A Stochastic Frame Based Approach to RFID Tag Searching

Ann L. Wang[1], Muhammad Shahzad[2], Alex X. Liu[1]
[1]CSE Dept., Michigan State University; [2]CS Dept., North Carolina State University
Email:{liyanwan, alexliu}@cse.msu.edu, mshahza@ncsu.edu

***Abstract*—This paper addresses the fundamental problem of RFID tag searching: given a set of known tag IDs and a population of RFID tags with unknown IDs, where the tags may be passive or active, we want to know which tag IDs are in the tag population. RFID tag searching has many applications such as product recall, inventory balancing, and stock verification. Previous RFID tag searching protocols cannot achieve arbitrarily high accuracy and are not C1G2 compliant. In this paper, we propose a protocol called RTSP, which satisfies the four requirements of C1G2 compliance, arbitrary accuracy, privacy preserving, and multiple-reader capability. RTSP is easy to deploy because it is implemented on readers as a software module and does not require any implementation on tags. Furthermore, it does not require any modifications either to tags or to the communication protocol between tags and readers and works with the commercially available off-the-shelf RFID tags. We implemented RTSP along with the fastest tag identification protocol and compared them side-by-side. Our experimental results show that RTSP always achieves the required accuracy and is 22.73% faster than the fastest RFID identification protocol.**

I. Introduction

A. Background and Motivation

As the cost of commercial RFID tags has become negligible compared to the prices of the products to which they are attached [1], RFID systems have been increasingly used in various applications such as supply chain management [2], indoor localization [3], inventory control, and access control [4]. For example, Walmart uses RFID tags to track expensive clothing merchandize [5] and Honeywell Aerospace uses RFID tags to track its products from birth to repair and retirement [6]. An RFID system consists of tags and readers. A tag is a microchip with an integrated antenna in a compact package that has limited computing power and communication range. There are two types of tags: passive tags and active tags. Passive tags do not have their own power source, are powered up by harvesting the radio frequency energy from readers, while the active tags have their own power sources. A reader has a dedicated power source with a significant amount of computing power. RFID systems work in a query-response fashion where a reader transmits queries to a set of tags and the tags respond with their IDs over a shared wireless medium.

This paper addresses the fundamental problem of RFID tag searching: *given a set of known tag IDs and a population of RFID tags with unknown IDs, where the tags may be passive or active, we want to know which tag IDs are in the tag population,* i.e., *search in a population of unknown tags for a set of known IDs*. RFID tag searching finds applications in product recall, inventory balancing, stock verification, and many other such settings. For product recall, if a manufacturer suspects that some of its products, which have already been distributed in different warehouses, are defective, they can use a tag searching protocol to quickly locate defective products, where the known tag IDs are defective products and the tag population are the products in a warehouse. For inventory balancing, if a large retailer, such as Amazon, wants to balance the quantity of different products among its warehouses across the country to reduce shipping time and costs, they can use a tag searching protocol to determine the quantity of any given product in each warehouse and then balance the quantity among warehouses accordingly, where the known tag IDs are the ones in inventory and the tag population are the ones in a warehouse. For stock verification, if a large retailer wants to check the quantity of each requested product sent to it in a large consignment, they can use a tag searching protocol to determine whether the consignment contains all requested products, where the known tag IDs are the ones that they are expecting and the tag population are the ones in the consignment. In this paper, we use the three terms, a tag, a tag ID, and the product that a tag is attached to, interchangeably.

B. Problem Statement

Now we formally define the tag searching problem. *Given a set A, which is a set of known tag IDs, a set B, which is a population of RFID tags with unknown IDs, a required confidence interval β, a tag searching protocol outputs $\tilde{C}$ so that $C \subseteq \tilde{C} \subseteq A$ and $|\tilde{C}| - |C| \leq \beta|C|$, where* $C = A \cap B$. Confidence interval β represents the maximum tolerable fraction of tags in A that are not in C but are declared as members of C by a tag searching protocol. A tag searching protocol should satisfy three additional requirements. First, it should comply with the EPCGlobal Class 1 Generation 2 (C1G2) RFID standard [7], which is a stable RFID standard and followed by the commercial RFID devices. Otherwise, it will be extremely difficult to be practically deployed.Second, it should preserve the privacy of the RFID tags in set B by not reading their tag IDs. Many RFID tag searching applications need to satisfy this privacy requirement. For example, if a policeman searches for some items with known tag IDs in a private house with a population of tags with unknown tag IDs, the home owner may prefer not to read the IDs of all tags in the house. Third, it should work with both a single-reader and multiple-reader environments. As the communication range

ISBN 978-3-901882-83-8 © 2016 IFIP

between a tag and a reader is limited, a large population of tags is often covered by multiple readers with overlapping regions.

C. Limitations of Prior Art

Previous RFID tag searching protocols (*i.e.*, [8]–[10] have two key limitations. First, they cannot achieve arbitrarily high accuracy. They are all probabilistic in nature, but none of them takes the confidence interval β as an input. Second, they do not comply with the C1G2 standard as they require the tags to receive, interpret, and act either according to pre-frame Bloom Filters or other protocol specific parameters. It is critical for RFID protocols to be compliant with the C1G2 standard because the cheap commercially available off-the-shelf (COTS) tags follow the C1G2 standard. A protocol that does not comply with the C1G2 standard will require custom tags, which will cost significantly more and have limited applications. Previous RFID identification protocols (such as TH [11], STT [12], MAS [13], and ASAP [14]) can be used to read all IDs of the tags in B and then calculate $C = A \cap B$. However, this straightforward solution has two key limitations. First, it does not preserve the privacy of the tags in B as it needs to read the IDs of all tags in B. Second, this is inefficient. We want an RFID tag searching protocol that is much faster than reading all tags in B.

D. Proposed Approach

In this paper, we propose a protocol called RFID Tag Searching Protocol (RTSP), which satisfies the following four requirement: (1) C1G2 compliance, (2) arbitrary accuracy, *i.e.*, $C \subseteq \tilde{C} \subseteq A$ and $|\tilde{C}| - |C| \leq \beta|C|$ for any required confidence interval β, (3) privacy preserving, and (4) multiple-reader capability.

To satisfy the requirement of C1G2 compliance, RTSP uses the frame slotted Aloha protocol specified in the C1G2 standard as its MAC layer communication protocol. In Aloha, the reader first tells the tags a frame size f and a random seed number R. Each tag within the transmission range of the reader then uses f, R, and its ID to select a slot in the frame by calculating a hash function $h(f, R, ID)$ whose result is uniformly distributed in $[1, f]$. Each tag has a counter initialized with the slot number that it chose to reply. After each slot, the reader first transmits an end of slot signal and then each tag decrements its counter by one. In any given slot, all the tags whose counters equal 1 respond with a random sequence called RN16. The reader uses this sequence to determine whether one or more than one tags are replying in that slot. If no tag replies in a slot, it is called an *empty slot*. If one or more tags reply in a slot, it is called a *nonempty slot*. Using 0 to denote an empty slot and 1 to denote a nonempty slot, after we execute the Aloha protocol on a population A of tags using frame size f and random seed R, we obtain a binary array of f bits, denoted as $\mathbb{S}(A, f, R)$.

To satisfy the requirement of arbitrary accuracy, RTSP executes n runs of the Aloha protocol where each run uses a different seed. For the i^{th} run with frame size f and random seed R_i, RTSP executes the Aloha protocol on both sets A and B, and thus obtains two binary arrays $\mathbb{S}(A, f, R_i)$ and $\mathbb{S}(B, f, R_i)$. Note that RTSP executes the Aloha protocol on A virtually as it knows all tag IDs in A. After n runs, for each tag ID $t \in A$, if for all $1 \leq i \leq n$, we have $\mathbb{S}(A, f, R_i)[h(f, R_i, t)] = \mathbb{S}(B, f, R_i)[h(f, R_i, t)]$, (*i.e.*, for all n runs, the two bits corresponding to tag t in both $\mathbb{S}(A, f, R_i)$ and $\mathbb{S}(B, f, R_i)$ are 1), then RTSP outputs $t \in \tilde{C}$. Clearly RTSP satisfies $C \subseteq \tilde{C} \subseteq A$. RTSP chooses a value of n so that $|\tilde{C}| - |C| \leq \beta|C|$.

To satisfy the requirement of privacy preserving, RTSP checks if a slot is empty or nonempty using the RN16 sequence and never asks tags to transmit their IDs. In C1G2, tags do not transmit their IDs unless the reader specifically asks them.

To satisfy the requirement of multi-reader capability, RTSP uses a central controller for all readers to use the same values for frame size f and seed R across all readers. The central controller uses a reader scheduling protocol [15] to ensure that two readers with overlapping regions do not transmit at the same time. When a reader transmits seed R_i in its i^{th} frame, it does not generate R_i on its own, rather, it uses the i^{th} seed R_i issued by the central controller. Thus, for a tag $t \in B$ that is covered by multiple readers, it chooses the same slot $h(f, R_i, t)$ for all readers. Once a reader completes its frame, it sends its binary array to the central controller. The controller applies the bit-wise logical OR operation on the binary arrays returned from all readers. The resulting binary array is the same as if there is one reader that covers all tags. RTSP uses this binary array to compute $\tilde{C}$.

E. Technical Challenges and Proposed Solutions

There are two key technical challenges in RTSP. The first technical challenge is to minimize tag searching time under the constraint that RTSP satisfies the required accuracy. To address this challenge, we use the accuracy requirement $|\tilde{C}| - |C| \leq \beta|C|$ to derive a *confidence condition*, which the system parameters such as frame sizes and execution rounds must satisfy. We then use the confidence condition to derive a *duration condition*, which system parameters must satisfy to minimize tag searching time. We then solve both conditions simultaneously to calculate the optimum system parameters that minimize tag searching time while achieving the required accuracy. The second technical challenge is to estimate the number of tags in set $|C|$, which is required to calculate the optimal values of system parameters. To address this challenge, RTSP counts the number of bits that are 1s in both $\mathbb{S}(A, f, R_i)$ and $\mathbb{S}(B, f, R_i)$. We call such bits dual-nonempty bits. The number of such dual-nonempty bits is a monotonically increasing function of $|C|$. By observing the number of dual-nonempty slots, RTSP estimates the value of $|C|$ while executing the Aloha protocol.

F. Advantages over Prior Art

The key novelty of this paper is in proposing a tag searching protocol that statistically guarantees to achieve any required accuracy and complies with the C1G2 standard. The key technical depth of RTSP lies in its mathematical development to guarantee any required accuracy and to minimize tag searching time. The key advantages of RTSP over prior tag searching protocols are that RTSP can achieve arbitrarily high accuracy and RTSP complies with the C1G2 standard. RTSP is easy to deploy because it is implemented on readers as a software module and does not require any implementation on tags. Furthermore, it does not require any modifications either to tags or to the communication protocol between tags and readers and works with the commercially available off-the-shelf RFID tags. RTSP can be implemented as a software module on readers. We have extensively evaluated the performance of RTSP. Our results show that for a scenario with $|A| = 5000$, $|B| = 5000$, and $|C| = 500$, and a required confidence interval of 0.1%, RTSP takes 15 seconds to search the tags whereas the fastest prior tag identification protocol (TH [11]) takes 22 seconds.

II. Related Work

To the best of our knowledge, there are four tag searching protocols [8]–[10], [16]. Zheng and Li proposed the first RFID tag searching protocol namely CATS [8]. CATS works in two phases. In the first phase, a server first constructs a Bloom filter by applying multiple hash functions in conjunction with a random seed on each tag ID in set A. Second, an RFID reader broadcasts the Bloom filter generated by the server along with the random seed to all tags in the population B. Using the received Bloom filter of set A, each tag in B checks if it is a candidate in C. Specifically, if all bits for a tag are 1s, the tag is a candidate in C; otherwise, it must be in $B - A$. Let B' denote all these candidates. Thus, due to false positives, $C \subseteq B' \subseteq B$. Then, the tags in B' distributively constructs another Bloom filter using the Framed Slotted Aloha protocol. The reader uses this Bloom filter to exclude the IDs in $A - B$. Thus, the reader obtains the searching result $A - (A - B) = A \cap B$. Unfortunately, C1G2 compliant tags can not interpret or generate Bloom filters, which makes CATS non-compliant with the C1G2 standard.

Chen *et al.* proposed another tag searching protocol called ITSP, which is an improved version of CATS [9]. In ITSP, the reader first generates a $k = 1$ Bloom filter on set A. Then, the reader broadcasts the Bloom filter along with the parameters used for constructing the Bloom filter to all tags in B. After a tag receives the Bloom filter, it checks whether it is in the Bloom filter. If a tag is in the Bloom filter, the tag will remain active; otherwise, it will become inactive. For the active tags, they collaboratively construct another $k = 1$ Bloom filter by executing the Framed Sloted Aloha protocol. ITSP repeats the above filtering process for multiple rounds until the false positive probability is below a certain threshold. Unfortunately, C1G2 compliant tags can not interpret or generate Bloom filters, which makes ITSP non-compliant with the C1G2 standard.

Zhang *et al.* proposed another tag searching protocol called TSM [10]. TSM extends CATS for use with multiple readers. It first executes CATS using each reader and then aggregates results from all readers to identify the tags in A that are present in B. Unfortunately, due to similar reasons as for CATS, TSM is also non-compliant with the C1G2 standard. In contrast, our proposed protocol, RTSP, is C1G2 compliant.

Liu *et al.* proposed BKC to count the number of tags in A that are present in B [16]. BKC first pre-computes a frame using IDs in set A and then executes a frame on population B to determine how many times the slots that were 1 in the pre-computed frame turned out to be 1 in the executed frame. It then uses the number of such slots to obtain the estimate of the number of tags in A that are present in B. BKC falls short because it can only estimate the number of tags in A that are present in B, but it can not determine exactly which tags of A are present in B. In contrast, our proposed protocol RTSP can identify such tags.

III. System Model

A. Architecture

For searching RFID tags, RTSP uses a central controller connected with a set of readers that cover the area where the tags in set B are located. The use of a central controller ensures that all readers use consistent values of frame sizes and seeds when executing frames, which helps in efficiently aggregating and processing information returned by the readers. The readers use the standardized frame slotted Aloha protocol to communicate with tags and never ask the tags to transmit their IDs. The use of multiple readers with overlapping coverage regions introduces following two problems: (1) scheduling the readers such that no two readers with overlapping regions transmit at the same time, and (2) alleviating the effect of some tags responding to multiple readers due to overlap in the coverage region of those readers. For the first problem, the controller uses one of the several existing reader scheduling protocols [15] to avoid reader-reader collisions. For the second problem, we propose solution in Section IV-A. RTSP does not require any modifications to tags or readers. It only requires the readers to receive system parameters from the controller and communicate the responses in the frames back to the controller.

B. C1G2 Compliance

RTSP does not require any modifications to tags or readers. It only requires the readers to receive the frame size, persistence probability, and seed number from the controller and communicate the responses in the frames back to the controller. Persistence probability p is the probability with which a tag decides whether it will participate in a frame or not before selecting a slot in that frame. Later in the paper, we will show how we use p to handle frame sizes that exceed the C1G2 specified upper limit of 2^{15}. Such large frame sizes are required when the size of tag population is large and required confidence interval β is small. With the use of p, the reader reduces the number of tags that participate in each frame, which in turn reduces the optimal frame size at the expense of increased number of frames. As the C1G2 standard does not specify the use of p, COTS tags do not support it. To avoid making any modifications to tags, in RTSP, the reader implements p by announcing a frame size of f/p but terminating the frame after the first f slots and sending a command to tags to reset their counters, which can be done as per the C1G2 standard.

C. Communication Channel

We assume that the communication channel between readers and tags is reliable *i.e.*, tags correctly receives queries from the readers and the readers correctly detect transmission of RN16 sequence in a slot if one or more tags in the population transmit in that slot. If the channel is unreliable, the solution proposed in [11] can be easily adapted for use with RTSP.

D. Independence Assumption

To make the formal development tractable, we assume that instead of picking a single slot to transmit at the start of i^{th} frame of size f, a tag independently decides to transmit in each slot of the frame with probability $1/f$ regardless of its decision about previous or forthcoming slots. Vogt first used this assumption for the analysis of Aloha protocol for RFID and justified its use by recognizing that this problem belongs to a class of problems called *occupancy problem*, which deals with the allocation of balls to urns [17]. Ever since, the use of this assumption has become a norm in the formal analysis of all Aloha based RFID protocols [17]–[19].

The implication of this assumption is that a tag can end up choosing more than one slots in the same frame or even not choosing any at all, which is not in accordance with the C1G2 standard that requires a tag to pick exactly one slot in a frame. However, this assumption does not create any problems because the expected number of slots that a tag chooses in a frame is still one. The analysis with this assumption is, therefore, asymptotically the same as that without this assumption [20]. Bordenave *et al.* further explained in detail why this independence assumption in analyzing Aloha based protocols provides results just as accurate as if all the analysis was done without this assumption [20]. This independence assumption is made only to make the formal development tractable. In our simulations, tag chooses exactly one slot at the start of frame.

IV. RFID Tag Search Protocol

A. Protocol Description

To search which tags in set A are present in the population B, in RTSP, the central controller executes n Aloha frames using the RFID readers. There are five steps involved in executing each frame. First, before executing any frame i, the controller calculates the optimal values of frame size f_i, persistence probability p_i, and generates a random seed number R_i. We will derive the expressions to calculate the values of f_i and p_i in the next section. Second, as the controller knows the IDs in set A, it virtually executes the Aloha protocol on set A and obtains the binary array $\mathbb{S}(A, f_i, R_i)$. Thus, the controller knows which bits in the binary array $\mathbb{S}(B, f_i, R_i)$ resulting from executing i^{th} frame on population B should be 1 if all the tags in A were present and a single reader covered the entire population. Third, it provides each reader with the parameters f_i, p_i, and R_i and asks each of them to execute the i^{th} frame using these parameters. The motivation behind using the same values of f_i, p_i, and R_i across all readers for the i^{th} frame is to enable RTSP to work with multiple readers with overlapping regions. As all readers use the same values of f_i, p_i, and R_i in the i^{th} frame, the slot number that a particular tag chooses in the i^{th} frame of each reader covering this tag is the same *i.e.*, $h(\frac{f_i}{p_i}, R_i, ID)$ evaluated by the tag results in same value for each reader. Fourth, each reader executes the frame on its turn as per the reader scheduling protocol and sends the responses in the frame back to the controller. Fifth, after the controller has received the i^{th} frame of each reader, it applies logical OR operator on all the received i^{th} frames and obtains the resultant bit array $\mathbb{S}(B, f_i, R_i)$. This resultant bit array $\mathbb{S}(B, f_i, R_i)$ is the same as if generated by a single reader covering all the tags. After obtaining the n bit arrays, $\mathbb{S}(B, f_i, R_i)$ for $1 \le i \le n$, for each tag $t \in A$, if $h(\frac{f_i}{p_i}, R_i, t) \le f_i$ the controller checks whether $\mathbb{S}(A, f_i, R_i)[h(\frac{f_i}{p_i}, R_i, t)] = \mathbb{S}(B, f_i, R_i)[h(\frac{f_i}{p_i}, R_i, t)]$ for all n frames, *i.e.*, for all n frames, whether the two bits corresponding to tag t in both $\mathbb{S}(A, f_i, R_i)$ and $\mathbb{S}(B, f_i, R_i)$ are 1s. If true, RTSP declares that the tag t is present in B. Note that RTSP can have false positives, *i.e.*, it can declare a tag in set A to be present in population B, when the tag is actually not present. RTSP does not have false negatives.

B. Estimating Number of Tags in Set C

Recall from the previous section that before executing any frame i, the controller calculates the optimal values of frame size f_i and persistence probability p_i. To calculate these optimal values for i^{th} frame, the controller needs estimate of $|C|$ at start of the i^{th} frame, which it obtains using the responses from the tag population in the previous $i-1$ frames. We represent the estimate of $|C|$ at the start of i^{th} frame by $|\tilde{C}_i|$. As the controller executes more and more frames, *i.e.*, as i increases, the estimate $|\tilde{C}_i|$ asymptotically becomes equal to $|C|$. Next, we present a method to estimate the value of $|C|$ at start of any frame i.

The intuition behind our estimation method is that as the number of tags in set C increases, the number of corresponding bits that are 1s in both $\mathbb{S}(A, f_i, R_i)$ and $\mathbb{S}(B, f_i, R_i)$ also increases. We call such bits as dual-nonempty bits. The number of dual-nonempty bits for any given frame is a function of $|C|$ and can, therefore, be used to estimate the value of $|C|$. Next, we derive an expression that relates the number of dual-nonempty bits with the value of $|C|$, *i.e.*, we derive an expression for $E[\mathcal{N}_i^{11}]$ as a function of $|C|$, where $\mathcal{N}_i^{11}$ is random variable for number of dual-nonempty bits in the pair of arrays $\mathbb{S}(A, f_i, R_i)$ and $\mathbb{S}(B, f_i, R_i)$. To derive the expression for $E[\mathcal{N}_i^{11}]$, we need the probability that any given pair of bits in the arrays $\mathbb{S}(A, f_i, R_i)$ and $\mathbb{S}(B, f_i, R_i)$ is dual-nonempty. We calculate this probability in the following lemma.

Lemma 1. *Let A be the set of IDs of tags that we want to search for in a population. Let B be the set of IDs of tags in the population in which we search for tags in set A. Let C be the set of IDs of those tags that are present in both sets A and B. Let X_{ij} be an indicator random variable for the event that the j^{th} bit in i^{th} pair of arrays is a dual-nonempty bit. For frame size f_i and persistence probability p_i, the probability distribution of X_{ij} is given by the following equation.*

$$P\{X_{ij}=1\}=1-(1-\frac{p_i}{f_i})^{|A|}-(1-\frac{p_i}{f_i})^{|B|}+(1-\frac{p_i}{f_I})^{|A|+|B|-|C|} \quad (1)$$

Proof. Probability that any given bit j in a pair of arrays is a dual-nonempty bit can be obtained by first calculating the probability that this bit is not a dual-nonempty bit, and then subtracting it from 1. The j^{th} bit is not dual-nonempty when one of the following three cases happens.

1) None of the tags in set A select the j^{th} slot in frame *i.e.*, the j^{th} bit in $\mathbb{S}(A, f_i, R_i)$ is 0, and none of the tags in population B select the j^{th} slot in corresponding executed frame *i.e.*, the j^{th} bit in $\mathbb{S}(B, f_i, R_i)$ is 0. We represent this event by an indicator random variable Y_{00}. The probability distribution of Y_{00} is given by the following equations.

$$P\{Y_{00}=1\} \quad = \quad \left(1-\frac{p}{f}\right)^{|A|+|B|-|C|} \quad (2)$$

2) One or more tags in set $A-C$ select the j^{th} slot in frame *i.e.*, the j^{th} bit in $\mathbb{S}(A, f_i, R_i)$ is 1, and none of the tags in population B select the j^{th} slot in corresponding executed frame *i.e.*, the j^{th} bit in $\mathbb{S}(B, f_i, R_i)$ is 0. We represent this event by an indicator random variable Y_{10}. The probability distribution of Y_{10} is given by the following equations.

$$P\{Y_{10}=1\} \quad = \quad \left(1-\left(1-\frac{p}{f}\right)^{|A-C|}\right)\left(1-\frac{p}{f}\right)^{|B|} \quad (3)$$

3) None of the tags in set A select the j^{th} slot in frame *i.e.*, the j^{th} bit in $\mathbb{S}(A, f_i, R_i)$ is 0, and one or more tags in population $B-C$ select the j^{th} slot in corresponding executed frame, *i.e.*, *i.e.*, the j^{th} bit in $\mathbb{S}(B, f_i, R_i)$ is 1. We represent this event by an indicator random variable Y_{01}. The probability distribution of Y_{01} is given by the following equations.

$$P\{Y_{01}=1\} = \left(1-\left(1-\frac{p}{f}\right)^{|B-C|}\right)\left(1-\frac{p}{f}\right)^{|A|} \quad (4)$$

The distribution of X_{ij} is given by the following equation.

$$P\{X_{ij}=1\} = 1-P\{Y_{00}=1\}-P\{Y_{10}=1\}-P\{Y_{01}=1\} \quad (5)$$

Substituting the expressions for the probability distributions of Y_{00}, Y_{10}, and Y_{01} from Equations (2) to (4), respectively, into Equation (5) and simplifying, we get Equation (1). □

Following theorem derives the expression for $E[\mathcal{N}_i^{11}]$ as a function of $|C|$.

Theorem 1. *Let A be the set of IDs of tags that we want to search for in a population. Let B be the set of IDs of tags in the population in which we search for tags in set A. Let C be the set of IDs of those tags that are present in both sets A and B. Let $\mathcal{N}_i^{11}$ be the random variable for the number of dual-nonempty bits in a pair of arrays of size f_i each. When persistence probability is p_i, the expected value of $\mathcal{N}_i^{11}$ is given by the following equation.*

$$E[\mathcal{N}_i^{11}] = f_i \times \left(1-\left(1-\frac{p_i}{f_i}\right)^{|A|}-\left(1-\frac{p_i}{f_i}\right)^{|B|} + \left(1-\frac{p_i}{f_i}\right)^{|A|+|B|-|C|}\right) \quad (6)$$

Proof. It is straight forward to see that $\mathcal{N}_i^{11}=\sum_{j=1}^{f_i} X_{ij}$. As $\{X_{i1}, X_{i2}, \ldots, X_{if_i)}\}$ forms a set of identically distributed random variables, $E[\mathcal{N}_i^{11}]$ is given by

$$E[\mathcal{N}_i^{11}] = E[\sum_{j=1}^{f_i} X_{ij}] = f_i \times E[X_{ij}]$$

As expected value of an indicator random variable equals its probability of being 1, $E[X_{ij}] = P\{X_{ij}=1\}$. Substituting value of $E[X_{ij}]$ in equation above with value of $P\{X_{ij}=1\}$ from Equation (5), we get the equation for $E[\mathcal{N}_i^{11}]$. □

Fig. 1 plots $E[\mathcal{N}_i^{11}]$ as a function of $|C|$ using Equation (6). This figure is obtained using $|A|=200$, $|B|=300$, $f_i=300$ and $p_i=1$. We observe from this figure that $E[\mathcal{N}_i^{11}]$ is a monotonically increasing function of $|C|$.

To estimate the value of $|C|$, let $\tilde{\mathcal{N}}_i^{11}$ represent the observed value of number of dual-nonempty bits for i^{th} pair of bit arrays. Replacing $E[\mathcal{N}_i^{11}]$ in Equation (6) with $\tilde{\mathcal{N}}_i^{11}$ and solving for $|C|$ gives an estimate of $|C|$. Using the well known identity $(1+x)^y \approx e^{xy}$ for small x and large y, Equation (6) can be written as follows.

$$E[\mathcal{N}_i^{11}] \approx f_i \times \left(1-e^{-\frac{p_i}{f_i}|A|}-e^{-\frac{p_i}{f_i}|B|}+e^{-\frac{p_i}{f_i}(|A|+|B|-|C|)}\right)$$

Replacing $E[\mathcal{N}_i^{11}]$ in the equation above with $\tilde{\mathcal{N}}_i^{11}$ and solving for $|C|$, we get the following equation to obtain the estimate $|\tilde{C}|$ of $|C|$.

$$|\tilde{C}| \approx |A|+|B|+\frac{f_i}{p_i}\ln\left\{\frac{\tilde{\mathcal{N}}_i^{11}}{f_i}-1+e^{-\frac{p_i}{f_i}|A|}+e^{-\frac{p_i}{f_i}|B|}\right\}$$

This estimate is obtained by utilizing the information from the i^{th} frame only. While this estimate may not be accurate, if we use the information from more frames, the estimate will become more accurate. Specifically, we leverage the well known statistical result that the variance in the observed value of a random variable reduces by x times if we take the average of x observations of that random variable. Therefore, to obtain the estimate $|\tilde{C}_i|$ of $|C|$ at the start of the i^{th} frame, we obtain an estimate from each of the previous $i-1$ frames and take their average. Solving Equation (6) for $|C|$ and averaging over past $i-1$ frames, the formal expression for $|\tilde{C}_i|$ becomes

$$|\tilde{C}_i| \approx |A|+|B|+\frac{\sum_{l=1}^{i-1}\frac{f_l}{p_l}\ln\left\{\frac{\tilde{\mathcal{N}}_l^{11}}{f_l}-1+e^{-\frac{p_l}{f_l}|A|}+e^{-\frac{p_l}{f_l}|B|}\right\}}{i-1} \quad (7)$$

Note that $|B|$ can be obtained using existing RFID estimation schemes such as ART [18]. Further note that the controller obtains this estimate without executing any additional frames. It gets this estimate from the frames it was already executing to search for tags.

V. Parameter Optimization

In this section, we will derive equations that the controller uses at the start of i^{th} frame to calculate the optimal values of frame size f_i and persistence probability p_i to minimize the execution time of RTSP while ensuring that its actual confidence interval is less than the required confidence interval. At the start of i^{th} frame, the controller uses the estimate $|\tilde{C}_i|$ along with the values of $|A|$, $|B|$, and β to calculate the optimal values of f_i and p_i. Before asking the readers to execute the i^{th} frame, the controller also calculates the minimum number of frames that it should execute, represented by n_i. Recall from

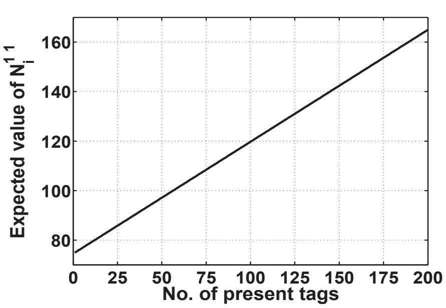


Fig. 1. $E[\mathcal{N}_i^{11}]$ vs. $|C|$

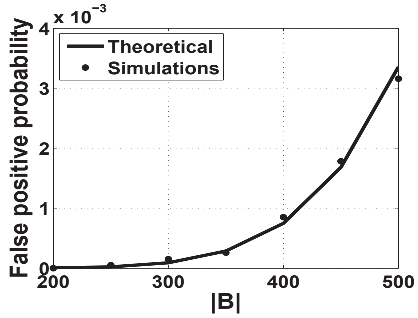


Fig. 2. Theoretical vs. experimental P_{fp}

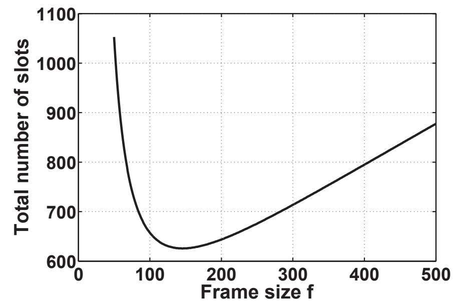


Fig. 3. Total number of slots S vs. frame size f

Section IV-B that as the number of executed frames increase, the estimate of $|C|$ becomes more accurate. Consequently, n_i, f_i, and p_i asymptotically become equal to constants n, f, and p, respectively. When the estimate of $|C|$ changes by less than 2 in 10 consecutive frames, the controller considers the estimate to be close enough to $|C|$. At this point, the controller calculates the values of n_i, f_i, and p_i one last time and puts $f = f_i$, $p = p_i$, and $n = n_i$, and uses these fixed values of f and p to execute subsequent frames until the total number of frames executed since the first frame become equal to n. For the first frame, *i.e.*, when $i = 1$, the controller uses $n_1 = \infty$, $f_1 = \max\{|A|, |B|\}$, and $p_1 = 1$. The choices of the values of n_1, f_1, and p_1 are arbitrary and do not really matter because as the controller executes more frames, number of frames, frame size, and persistence probability converge to constants n, f, and p, respectively.

In subsequent calculation of n_i, f_i, and p_i, we will drop the subscript i to make the presentation simple. Next, we first derive the expression for false positive probability *i.e.*, probability with which RTSP declares a tag in set A to be present in population B, when it actually is not. Second, using the expression for false positive probability, we derive a *confidence condition* that the values of n, f, and p must satisfy to ensure that the observed confidence interval is smaller than the required confidence interval β, *i.e.*, the requirement $|\tilde{C}| - |C| \leq \beta|C|$ is satisfied. Third, we derive a *duration condition*, which the values of f and p must satisfy to ensure that the execution time of RTSP is minimized. The controller solves these two conditions simultaneously to obtain the optimal values of n, f, and p. Last, we describe our strategy to bring the value of f within limit when the optimal frame size exceeds the C1G2 specified upper limit of 2^{15}.

A. False Positive Probability

A false positive occurs when all the bits that a particular tag in A that is not present in B selects in the n bit arrays $\mathbb{S}(A, f_i, R_i)$ for $1 \leq i \leq n$, turn out to be nonempty in corresponding bit arrays $\mathbb{S}(B, f_i, R_i)$ because some other tags in the population made those bits 1. Lemma 2 gives the expression to calculate the false positive probability.

Lemma 2. *Let B be the set of IDs of tags in the population in which we search for tags. With persistence probability p, frame size f, and number of frames n, the false positive probability, P_{fp}, is given by* $P_{fp} = \left[1 - \left(1 - \frac{p}{f}\right)^{|B|}\right]^n$.

Proof. Consider a tag t such that $t \in A \wedge t \notin B$. The probability that the bit tag t selects in $\mathbb{S}(A, f_i, R_i)$ is selected by at least one tag in population B in $\mathbb{S}(B, f_i, R_i)$ is $1-(1-\frac{p}{f})^{|B|}$. The probability that all n bits tag t selects in the n bit arrays $\mathbb{S}(A, f_i, R_i)$ for $1 \leq i \leq n$, are also selected by some other tags in population B in corresponding bit arrays $\mathbb{S}(B, f_i, R_i)$ is $[1-(1-\frac{p}{f})^{|B|}]^n$, which is the expression for P_{fp}, given in the lemma statement. □

Figure 2 shows the theoretically calculated false positive probability from equation of P_{fp} in Lemma (2) represented by the solid line and experimentally observed values of false positive probability represented by the dots. To obtain this figure, we use $f = 600$, $p = 1$, and $n = 10$. Each dot represents the false positive probability calculated from 200 runs of simulation. We observe that the theoretically calculated values match perfectly with experimentally observed values, showing that our independence assumption that we stated in Section III-D does not cause the theoretical analysis to deviate from practically observed values.

B. Confidence Condition

Theorem 2 states the confidence condition, which the values of n, f, and p must satisfy to achieve the required confidence interval β.

Theorem 2. *Let A be the set of IDs of tags that we want to search for in a population. Let B be the set of IDs of tags in the population in which we search for tags in set A. Let C be the set of IDs of those tags that are present in both sets A and B. To ensure that RTSP satisfies the requirement $|\tilde{C}| - |C| \leq \beta|C|$, the controller must use the values for number of frames n, frame size f, and persistence probability p that satisfy the confidence condition given in the following equation.*

$$n = \frac{\ln\left(\frac{\beta \times |\tilde{C}|}{|A| - |\tilde{C}|}\right)}{\ln\left(1 - (1 - \frac{p}{f})^{|B|}\right)} \quad (8)$$

Proof. Let $E[|\tilde{C}|]$ represent the number of tags that RTSP declares as belonging to set C after executing n frames of size f with persistence probability p. Replacing $|\tilde{C}|$ in $|\tilde{C}| - |C| \leq \beta|C|$ by $E[|\tilde{C}|]$, the confidence requirement is given by $E[|\tilde{C}|] - |C| \leq \beta|C|$. Next, we derive the expression for $E[|\tilde{C}|]$. Recall from Section IV-A that RTSP can have false positives, but it cannot have false negatives *i.e.*, it will always identify the tags of A present in B and in addition, it may also declare some tags in A that are not in B to be present in B. Thus, $E[|\tilde{C}|] = |C| + (|A - C|) \times P_{fp}$. As $C \subseteq A$, thus, $E[|\tilde{C}|] = |C| + (|A| - |C|) \times P_{fp}$ Substituting this value of $E[|\tilde{C}|]$ into the confidence requirement, we get the following equation: $|C|+(|A|-|C|)\times P_{fp}-|C| \leq \beta|C|$. Substituting the value of P_{fp} from Lemma 2 into this equation and rearranging, we get $n \geq \frac{\ln\left(\frac{\beta\times|C|}{|A|-|C|}\right)}{\ln\left(1-(1-\frac{p}{f})^{|B|}\right)}$. As we do not know the exact value of $|C|$, rather we know the estimate $|\tilde{C}|$ of $|C|$, replacing $|C|$ in this equation with $|\tilde{C}|$ and using the smallest value for n allowed by the equation above to ensure that confidence requirement is always met, we get Equation (8) in theorem statement. □

C. Duration Condition

Theorem 3 states the duration condition that the values of f and p must satisfy to minimize the execution time of RTSP.

Theorem 3. *Let A be the set of IDs of tags that we want to search for in a population. Let B be the set of IDs of tags in the population in which we search for tags in set A. Let C be the set of IDs of those tags that are present in both sets A and B. To ensure that the execution time of RTSP is minimum, the controller must use the values for frame size f and persistence probability p that satisfy the duration condition given in the following equation.*

$$p \times |B| = f \times \left(1 - e^{\frac{p}{f}|B|}\right) \times \ln\left\{1 - e^{-\frac{p}{f}|B|}\right\} \quad (9)$$

Proof. Execution time is directly proportional to the total number of slots because the duration of each slot is the same, typically $300\mu s$ for Philips I-Code RFID reader [21]. Let S represent the total number of slots. Thus, $S = f \times n$. To ensure that RTSP achieves the required confidence interval, we use the value of n from Equation (8). Thus,

$$S = \frac{f \ln\left(\frac{\beta\times|\tilde{C}|}{|A|-|\tilde{C}|}\right)}{\ln\left(1-(1-\frac{p}{f})^{|B|}\right)} \quad (10)$$

Figure 3 plots S as a function of f using the equation above. This figure is made using $|A| = 100$, $|B| = 100$, $|\tilde{C}| = 52$, $p = 1$, and $\beta = 0.05$. We observe from this figure that S is a convex function of f. Therefore, optimum value of f exists, represented by f_{op}, that minimizes the total number of slots S. To find optimal value of f, we differentiate the equation above w.r.t f and equate the resulting expression to 0, and get the following:

$$\left[\ln\left(\frac{\beta\times|\tilde{C}|}{|A|-|\tilde{C}|}\right)\right]\left[p|B| - f\left(1 - e^{\frac{p}{f}|B|}\right)\ln\left\{1 - e^{-\frac{p}{f}|B|}\right\}\right] = 0$$

Note that $\ln\left(\frac{\beta\times|\tilde{C}|}{|A|-|\tilde{C}|}\right) \neq 0$, which means that the following must hold true: $p|B| - f\left(1 - e^{\frac{p}{f}|B|}\right)\ln\left\{1 - e^{-\frac{p}{f}|B|}\right\} = 0$ Rearranging the equation above, we get the duration condition in the theorem statement. □

The controller solves Equations (8) and (9) simultaneously using $p = 1$ and gets the optimal values of n and f represented by n_{op} and f_{op}, respectively. It calculates f_{op} numerically from Equation (9) using Brent's method. Then it puts $f = f_{\text{op}}$ and $p = 1$ in Equation (8) to calculate n_{op}. Next, we study the effect of $|A|$, $|B|$, $|C|$, and β on execution time of RTSP.

Execution Time vs. $|A|$: Intuitively, as the number of tags in A increases, the execution time of RTSP should increase because the greater number of tags in A implies the higher chances of false positives. Thus, to ensure that the number of false positives stays small enough so that the required confidence interval is achieved, RTSP executes more frames, *i.e.*, the value of n_{op} increases, which increases the overall execution time. Figure 4(a) confirms our intuition. This figure plots the expected execution time of RTSP for multiple values of $|A|$ while fixing $|B|$ at 5000 and $|C|$ at 500. We calculated the execution time as $n_{\text{op}} \times f_{\text{op}} \times T_s$, where T_s is the time of each slot and is equal to $300\mu s$ as per the specifications of Philips I-Code RFID reader [21]. We observe from Figure 4(a) that as the number of tags in A increases, the execution time of RTSP increases. The stairway behavior that RTSP shows in this and subsequent figures is due to the ceiling operation on the non-integer values of n_{op} and f_{op}.

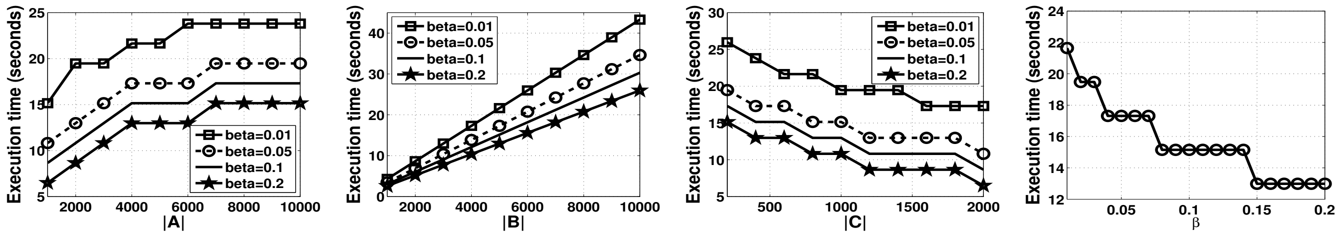
Execution Time vs. $|B|$: Intuitively, as the number of tags in B increases, the execution time of RTSP should increase because greater number of tags in B also imply higher chances of false positives. Thus, to ensure that the number of false positives stays small enough so that the required confidence interval is achieved, RTSP increases the frame size, *i.e.*, the value of f_{op} increases according to Equation (9), which increases the overall execution time. Figure 4(b) confirms our intuition. This figure plots the expected execution time of RTSP for multiple values of $|B|$ while fixing $|A|$ at 5000 and $|C|$ at 500. We observe from Figure 4(b) that as the number of tags in B increases, the execution time of RTSP increases.

Execution Time vs. $|C|$: Intuitively, as the number of tags in C increases, the execution time of RTSP should decrease because greater number of tags in C means RTSP has greater margin of error *i.e.*, $\beta|C|$. Thus, RTSP reduces the value of n_{op}, which decreases the overall execution time. Figure 4(c) confirms our intuition. This figure plots the expected execution time of RTSP for multiple values of $|C|$ while fixing $|A|$ at 5000 and $|B|$ at 5000. We observe from Figure 4(c) that as the number of tags in C increases, the execution time of RTSP decreases.

Execution Time vs. β: Intuitively, as the required confidence interval β increases, the execution time of RTSP should decrease because larger required confidence interval means RTSP has greater margin of error. Thus, RTSP reduces the values of n_{op}, which decreases the overall execution time. Figure 4(d) confirms our intuition. This figure plots the expected execution time of RTSP for different values of β while fixing $|A|$ at 5000, $|B|$ at 5000, and $|C|$ at 500. We observe from Figure 4(d) that as the required confidence interval increases, the execution time of RTSP decreases.

D. Handling Large Frame Sizes

For large populations and/or small required confidence interval, it is possible for the value of f_{op} to exceed the C1G2 specified upper limit of 2^{15}. Next, we describe how we use p



(a) Execution time vs. $|A|$ (b) Execution time vs. $|B|$ (c) Execution time vs. $|C|$ (d) Execution time vs. β

Fig. 4. Effect of $|A|$, $|B|$, $|C|$, and β on execution time of RTSP

to bring the frame size within limits. Bringing the frame size within limits comes at a cost of increased number of slots; greater than the minimum value of S that would have been achieved if the controller could use $f_{\text{op}} > 2^{15}$.

When we decrease the value of p, the number of tags that participate in a frame decreases. Therefore, the required value of f also decreases. Participation by fewer tags means that participation by the tags belonging to both the sets A and B decreases. This increases the chances that a given tag in A that is present in B will not select any slot in a given pre-computed frame, which means that chances of identifying its presence decrease. Therefore, the overall uncertainty in identifying tags in A increases. To reduce this uncertainty, the n increases when p decreases to achieve the required confidence interval.

We use these two observations to reduce the value of f whenever $f_{\text{op}} > 2^{15}$. When $f_{\text{op}} > 2^{15}$, the controller uses $f = f_{\max} = 2^{15}$ in Equation (8), which leaves two unknowns, p and n, in the resulting equation. The controller solves the resulting equation simultaneously with Equation (9) to get new values of p and n. The new value of p is less than 1 and the new value of n is greater than n_{op} (we represent n with n_{op} only when we use $f = f_{\text{op}}$ to calculate it). The controller uses these new values of n and p along with $f = f_{\max}$ to compute the bit array $\mathbb{S}(A, f_i, R_i)$. Although the total number of slots $S = f_{\max} \times n > f_{\text{op}} \times n_{\text{op}}$, this is still the smallest under the constraints that the required confidence interval is achieved and the frame size does not exceed $f_{\max}$.

VI. Performance Evaluation

We implemented and simulated RTSP in Matlab. We also implemented and simulated the fastest existing tag identification protocol, TH [11], to compare the execution time of RTSP with it. We choose tag ID length of 64 bits as specified in the C1G2 standard. Note that the distributions of the IDs of tags in A and B do not matter because RTSP is independent of ID distributions. Next, we first evaluate the accuracy of RTSP and then compare its execution time with the execution time of TH. All results reported in this section are obtained from averaging over 200 independent runs of RTSP.

A. Accuracy

To evaluate the accuracy of RTSP, we study its confidence interval for different values of $|A|$, $|B|$, and $|C|$.

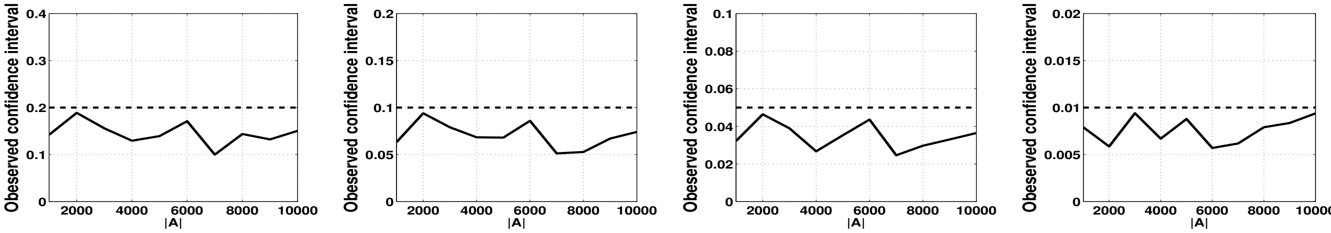
1) Observed Confidence interval vs. $|A|$: Our experimental results show that RTSP always achieves the required confidence interval regardless of the size of set A. Figures 5(a), 5(b), 5(c), and 5(d) plot the actual confidence interval RTSP achieved for different sizes of set A when the required values of confidence interval are $\beta = 0.2$, $\beta = 0.1$, $\beta = 0.05$, $\beta = 0.01$, respectively. To plot these figures, we fixed number of tags in set B at 5000 and number of tags in A that are in B, *i.e.*, number of tags in set C at 500. The dashed horizontal line in each of these figures shows the required value of confidence interval and the solid line shows the observed values of confidence interval achieved by RTSP. We observe from these figures that the observed values of confidence interval are always smaller than the required values of confidence interval.

2) Observed Confidence interval vs. $|B|$: Our experimental results show that RTSP always achieves the required confidence interval regardless of the number of tags in population B. We have not included figures due to lack of space.

3) Observed Confidence interval vs. $|C|$: Our experimental results show that RTSP always achieves the required confidence interval regardless of the number of tags in set C. Figures 6(a), 6(b), 6(c), and 6(d) plot the actual confidence interval RTSP achieved for different sizes of set C when the required values of confidence interval are $\beta = 0.2$, $\beta = 0.1$, $\beta = 0.05$, $\beta = 0.01$, respectively. To plot these figures, we fixed number of tags in sets A and B at 5000 each. Again, we observe from these figures that the solid lines are always below their corresponding dashed lines, which means that RTSP always achieves the required confidence interval.

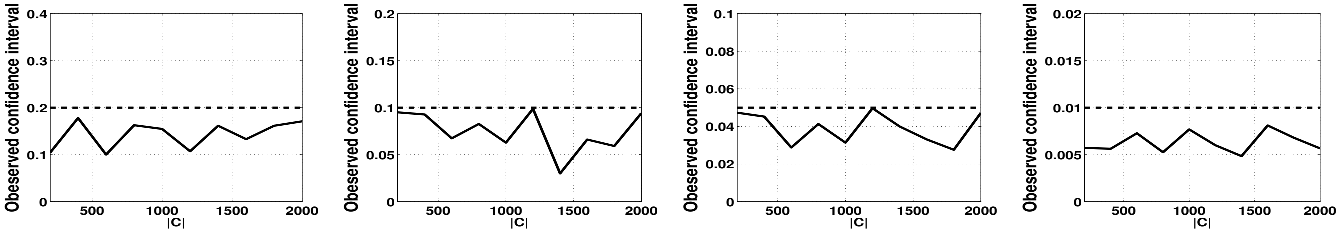
B. Execution Time

Execution time of RTSP is smaller than TH. Figure 7(a) plots the execution times of TH and RTSP vs. $|A|$ for $\beta = 0.1$, $|B| = 3000$, and $C = 500$. We observe from this figure that RTSP is up to 22.73% faster compared to TH. Similarly, Figure 7(b) plots the execution times vs. $|B|$ for $\beta = 0.1$, $|A| = 1000$, and $|C| = 500$ and Figure 7(c) plots the execution times vs. $|C|$ for $\beta = 0.1$, $|A| = 5000$, and $|B| = 5000$. Again, we observe from these figures that RTSP is always faster compared to TH. Finally, Figure 7(d) plots the execution times vs. β for $|A| = 5000$, $|B| = 5000$, and $|C| = 500$. We observe that RTSP is faster compared to TH as long as required confidence interval is > 0.01. When the required confidence interval < 0.01, TH is faster. Thus, if privacy is not a concern, a user should use TH whenever $\beta < 0.01$. If, however, privacy is a concern, the user should always use RTSP.



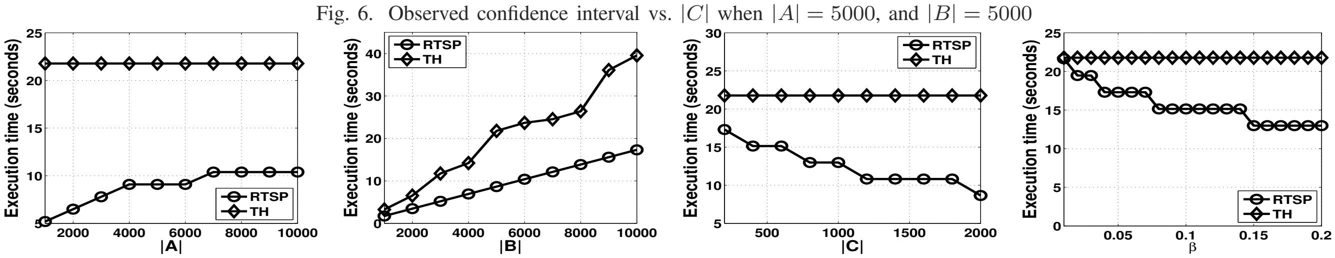
(a) required $\beta = 0.2$ (b) required $\beta = 0.1$ (c) required $\beta = 0.05$ (d) required $\beta = 0.01$

Fig. 5. Observed confidence interval vs. $|A|$ when $|B| = 5000$, and $|C| = 500$



(a) required $\beta = 0.2$ (b) required $\beta = 0.1$ (c) required $\beta = 0.05$ (d) required $\beta = 0.01$

Fig. 6. Observed confidence interval vs. $|C|$ when $|A| = 5000$, and $|B| = 5000$



(a) Execution time vs. $|A|$ (b) Execution time vs. $|B|$ (c) Execution time vs. $|C|$ (d) Execution time vs. β

Fig. 7. Comparison of execution times of RTSP and TH

VII. Conclusion

The key technical contribution of this paper is in proposing a protocol to search tags in a population of RFID tags. This paper represents the first effort on addressing this important and practical problem for C1G2 compliant RFID systems. The key technical depth of this paper is in the mathematical development of the theory that RTSP is based on. The solid theoretical underpinning ensures that RTSP always achieves the required confidence interval. We have proposed a technique to handle large frame sizes to ensure the compliance with the C1G2 standard. We have also proposed a method to implicitly estimate the number of tags in set C.We implemented RTSP and conducted side-by-side comparisons with TH, the fastest prior tag identification protocol. Our experimental results show that RTSP always achieves the required confidence interval and significantly outperforms TH in terms of search time.

References

[1] M. Roberti, "A 5-cent breakthrough," *RFID Journal*, vol. 5, no. 6, 2006.

[2] C. H. Lee and C.-W. Chung, "Efficient storage scheme and query processing for supply chain management using RFID," in *Proc. ACM Conf. on Management of data*, pp. 291–302, 2008.

[3] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: Indoor location sensing using active RFID," *Wireless networks*, vol. 10, 2004.

[4] K. Finkenzeller, *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication*. Wiley, 2010.

[5] M. Roberti, "Wal-mart relaunches EPC RFID effort, starting with men's jeans and basics," *RFID Journal*, 2010.

[6] Swedberg, "Honeywell aerospace tags parts for airbus," *RFID Journal*.

[7] E. Inc, *Radio-Frequency Identity Protocols Class-1 Generation-2 Protocol for Communications at 860 MHz–960 MHz*. EPCGlobal Inc.

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

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

[10] S. Zhang, X. He, H. Song, and D. Zhang, "Time efficient tag searching in multiple reader RFID systems," in *Proc. Green Computing and Communications (GreenCom)*, IEEE, 2013.

[11] M. Shahzad and A. X. Liu, "Probabilistic optimal tree hopping for RFID identification," in *Proc. ACM SIGMETRICS*, 2013.

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

[13] V. Namboodiri and L. Gao, "Energy-aware tag anticollision protocols for RFID systems," in *Proc. 5th IEEE Int. Conf. on Pervasive Computing and Communications*, pp. 23–36, 2007.

[14] C. Qian, Y. Liu, H. Ngan, and L. M. Ni, "ASAP: Scalable identification and counting for contactless RFID systems," in *Proc. 30th IEEE Int. Conf. on Distributed Computing Systems*, pp. 52–61, 2010.

[15] S. Tang, J. Yuan, X.-Y. Li, G. Chen, Y. Liu, and J. Zhao, "Raspberry: A stable reader activation scheduling protocol in multi-reader RFID systems," in *Proc. IEEE ICNP*, 2009.

[16] X. Liu, K. Li, H. Qi, B. Xiao, and X. Xie, "Fast counting the key tags in anonymous RFID systems," in *Proc. IEEE ICNP*, 2014.

[17] H. Vogt, "Efficient object identification with passive RFID tags," *Pervasive Computing*, vol. 2414, pp. 98–113, 2002.

[18] M. Shahzad and A. X. Liu, "Every bit counts: fast and scalable RFID estimation," in *Proc. ACM MobiCom*, 2012.

[19] B. Zhen, M. Kobayashi, and M. Shimizu, "Framed ALOHA for multiple RFID objects identification," *IEICE Transactions on Communications*, vol. 88, pp. 991–999, 2005.

[20] C. Bordenave, D. McDonald, and A. Proutiere, "Performance of random medium access control, an asymptotic approach," in *Proc. ACM SIGMETRICS*, 2008.

[21] P. Semiconductors, *http://www.advanide.com/datasheets/sl2ics11.pdf*.