Fast Splitting Based Tag Identification Algorithm For Anti-collision in UHF RFID System

Jian Su, Zhengguo Sheng, Liangbo Xie, and Gang Li

***Abstract*—Efficient and effective objects identification using Radio Frequency Identification (RFID) is always a challenge in large scale industrial and commercial applications. Among existing solutions, the tree based splitting scheme has attracted increasing attention because of its high extendibility and feasibility. However, conventional tree splitting algorithms can only solve tag collision with counter value equals to zero and usually result in performance degradation when the number of tags is large. To overcome such drawbacks, we propose a novel tree-based method called Fast Splitting Algorithm based on Consecutive Slot Status detection (FSA-CSS), which includes a fast splitting (FS) mechanism and a shrink mechanism. Specifically, the FS mechanism is used to reduce collisions by increasing commands when the number of consecutive collision is above a threshold. Whereas the shrink mechanism is used to reduce extra idle slots introduced by FS. Simulation results supplemented by prototyping tests show that the proposed FSA-CSS achieves a system throughput of 0.41, outperforming the existing UHF RFID solutions.**

***Index Terms*—RFID, UHF, anti-collision, FS mechanism, system throughput, time efficiency.**

I. Introduction

RFID technology has greatly revolutionized tag based applications in retail industry such as warehouse management and inventory control [1-2]. A typical RFID system is composed of a reader and a large number of tags attached to tracked objects. Each object can be identified based on query response from the attached RFID tag. However, in large scale RFID applications, simultaneous query responses from multiple tags can cause significant tag collisions. Since passive tags are unable to perceive or identify such collisions, the development of anti-collision algorithms is of great importance for fast tag identification especially in high-density ultra high frequency (UHF) RFID environment.

Existing tag anti-collision algorithms can be categorized into dynamic framed slotted Aloha (DFSA)[3-5], query tree (QT)[6-8] and binary splitting (BS) based [9-10] algorithms. Among them, QT and BS algorithms are operated by recursively dividing responding tags into smaller subsets until each subset has at most one tag. The distinction between these two approaches is that, in QT solutions, tags are separated by their IDs. Whereas in BS approaches, tags are divided by binary random numbers generated in the splitting process. Strictly speaking, QT methods are derived from the bit tracking technology [11] which can detect the position of collided bit by the reader. However, with an increasing number of tags, the efficiency of such methods is deteriorated because of the wide deviation of backscatter link frequency among tags [12-14].

As a contrary, DFSA and BS algorithms are more preferable for UHF RFID systems. DFSA algorithms usually employ a frame structure which contains a certain number of time intervals (called time slots) per frame, and each tag responses to the reader by randomly choosing a time slot using its ID. During the identification process, the size of frame is dynamically updated according to the number of unread tags. When the frame size is equal to the backlog (number of unread tags), the maximum system throughput can be achieved. Recent works in DFSA include improved linearized combinatorial model (ILCM) [15] based anti-collision algorithm, an efficient anti-collision algorithm with early adjustment of frame length (EAAEA) [16], and access probability adjustment based fine-grained Q-algorithm (APAFQ) [17], etc. However, those algorithms have failed to prevent collisions completely, because of the tag starvation problem in which a specific tag may not be identified for a long time [9]. Furthermore, the performance of DFSA algorithms highly depends on the initial frame size. When the number of tags is much larger than the size of frame, most of DFSA solutions are unable to adjust frame size properly in order to cope with backlog, thus lead to performance degradation [3, 15]. That is to say, DFSA solutions are cardinality sensitive and shows inconsistent performance with a wide range of backlog. As a contrary, BS algorithms are insensitive to tag backlog particularly when the number of tags is increased, the system throughput is almost converged to a constant value. Although the BS approach can tackle tag starvation, it has a relatively long identification latency due to the splitting process starting from a single set with all tags. Specially, the system throughput of BS algorithms is about 0.348 [10] when the number of tags is larger than 100. Most of recent anti-collision algorithms with high performance are based on the integration of Aloha and QT (or BS) algorithms [18-19]. These methods usually need to estimate cardinality of tag population. Many efforts have also been made to improve estimation accuracy [3-4][15][20]. However, constant estimation of number of tages with high accuracy requires high computation overhead and thus leads to serious challenges in implementation [16].

In this paper, we focus on the UHF RFID anti-collision algorithm and propose a fast splitting algorithm based on

J. Su is with Nanjing University of Information Science and Technology, Jiangsu 210044, China (e-mail: sj890718@gmail.com).

Z. Sheng (corresponding author) is with the Department of Engineering and Design, University of Sussex, Brighton BN1 9RH, UK (e-mail: z.sheng@sussex.ac.uk).

L. Xie is with Chongqing University of Posts and Telecommunications, Chongqing 400065, China (e-mail: xie.liangbo@hotmail.com).

G. Li is with University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: ligang1986718@163.com).

Digital Object Identifier xxxx

consecutive slot status detection (FSA-CSS) to improve the identification and stability performance of BS algorithm. The proposed solution is based on the pure BS algorithm which is rarely investigated given concerns of marginal performance improvement by the existing literature. Different to the existing solutions in which the average performance is only investigated, i.e., incoming and outgoing tags are allowed to be identified multiple times [21-22], we focus on a practice-driven and challenging scenario in which a tag can only be identified once and propose to improve individual performance for identifying a batch of tags. It shows from the results that FSA-CSS can still maintain high performance in large-scale RFID systems without estimation of backlog. The contributions of this paper can be summarized as follows.

1) We propose an enhanced anti-collision algorithm namely FSA-CSS for passive UHF RFID systems. In order to accelerate the splitting process, the reader allows the tags with counter value above zero to be split into subsets in advance. Meanwhile, the reader can avoid over-splitting by using the shrink mechanism to reduce the extra idle slots.

2) We provide the theoretical analysis and carry out simulations to evaluate the performance of FSA-CSS with a massive number of tags. The results have been compared with various legacy anti-collision algorithms. The optimal parameters and performance boundaries have been derived.

3) We implement FSA-CSS in a practical UHF RFID system, which includes a passive RFID reader and 20 tags. The identification time of the proposed FSA-CSS is reduced by 32.5% compared to the standard BS used in ISO/IEC 18000-6B.

The rest of this paper is organized as follows. Section II reviews and analyzes the mainstream tag identification strategies for UHF RFID systems. Section III discusses the novel anti-collision algorithm FSA-CSS and analyzes its performance. Section IV illustrates the simulation results. The experimental results are presented in Section V. Finally, the paper is concluded in Section VI.

II. Related Works

A. Binary splitting (BS) algorithm

The principle of BS algorithm is to continuously divide collided tags into smaller subsets by using random binary numbers. Each tag has an integer counter T_c which is used to record its position in the splitting process and a random binary number generator T_R. Different values of T_c lead tags into different subsets. Only tags with $T_c = 0$ respond to the reader immediately, while tags with $T_c>0$ should wait in the pipeline. At the initial stage of identification process, all tags in reader's vicinity should respond simultaneously and thus are formed as one set. Depending on received responses, the reader will send a feedback, e.g., `ID-idle`, `ID-collision` or `ID-success`, to all tags for further actions. When the feedback is `ID-success` or `ID-idle`, all tags act $T_c = T_c - 1$. The tags already identified by the reader will be silent during the next identification process. If the feedback is `ID-collision`, the collided tags will be divided into two subsets. Tags with $T_c = 0$ add T_R to its T_c, while tags with $T_c>0$ increase its T_c by 1.

The reader also uses a counter R_c to terminate the identification process. R_c denotes the number of unidentified tags and its initial value is 0. When a collision occurs, the reader performs $R_c = R_c + 1$, given the number of identifiable tags is increased. Otherwise, R_c is decreased by 1. When $R_c < 0$, the identification process is ceased. Compared to Aloha-based algorithms, the BS algorithm is insensitive to tag cardinality particularly when the number of tags is increased. The system throughput is almost converged to a constant value. The work in [10] reveals that the system throughput of BS can be maintained at 0.348 when the number of tags is above 100. However, the BS algorithm has a relatively long identification latency because the splitting process is started from a single set with all tags. Moreover, it always use tag ID to perform collision arbitration, hence there is significant space to further improve time efficiency.

B. DFSA protocol with backlog estimation

In DFSA protocols, time is divided into a series of discrete intervals called time slots. Several time slots are packaged into a frame [3][15]. Each tag can randomly select a slot to respond to the reader in each identification round. At the end of each frame, the reader counts the number of idle, success, and collision slots. If collision slots exist, the reader will estimate the number of unidentified tags and adjust the frame length accordingly. The identification process continues until no collision occurs during a frame, i.e., all tags are successfully identified.

It is noted that the reader needs to accurately estimate the backlog to achieve the best performance. To improve estimation accuracy, most previous methods [3-5] are implemented with high complexity. However, typical RFID readers are computation constrained due to the limited processing power provided by single-chip microprocessor. As a result, estimation methods with high computation overhead are characterized as energy inefficient. Recently, many state-of-the-art algorithms [15-16] have been proposed to achieve energy efficiency. However, the system throughput of these solutions is still below 0.36. In addition, it is worth mentioning that most of existing DFSA studies are simulation based [3-5][15-17], thus the practical performance cannot be verified.

C. Hybrid protocol combining DFSA with BS

So far it has been shown that the BS performs better when the number of tags is relatively small [10]. This result has inspired further studies to design hybrid anti-collision algorithms combining DFSA and BS. The authors in [23] propose an adaptive binary tree slotted Aloha (ABTSA) algorithm. In ABTSA, tags are randomly assigned to slots in a frame. If multiple tags are collided in a slot, collision will be solved by the BS algorithm while other tags in the following slots will be in waiting state. The benefit of using ABTSA is that the adjustment of frame size is simplified because the reader has the knowledge of the status in every slot and thus can obtain an appropriate frame size accordingly. Since the ABTSA combines both DFSA and BS algorithms, it can achieve an average system throughput of 0.40, which is higher than that of DFSA or BS algorithm alone. However, it is with a complex structure, hence much more difficult to be implemented.

In summary, aforementioned anti-collision algorithms do provide solutions to solve the tag collision problem, however, with sacrifices in identification efficiency, complexity and stability, etc. The characteristics of existing anti-collision algorithms are summarized in Tab. I. In the following, we will introduce the proposed solution based on the BS algorithm to improve the identification efficiency of RFID system and reduce the implementation cost.

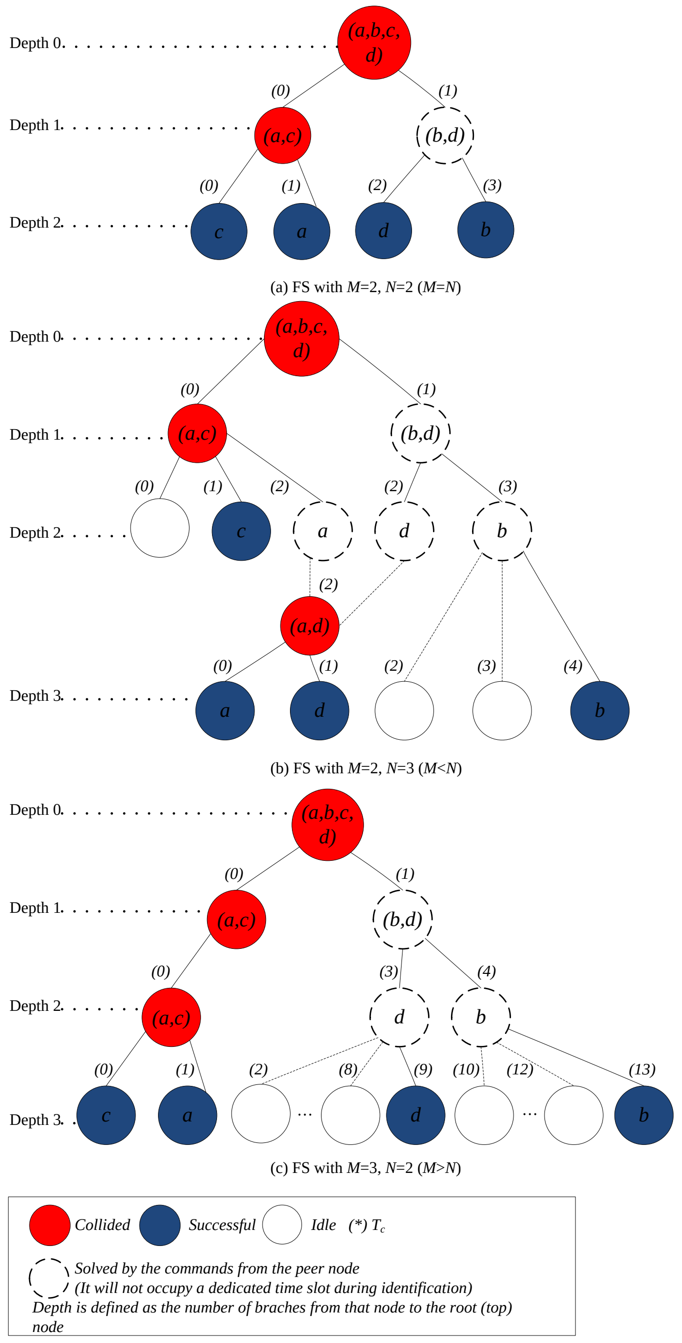
Depth 0
Depth 1
Depth 2
Depth 3
(a,b,c, d)
(a,c)
(b,d)
(a,d)
a
b
c
d
(0)
(1)
(2)
(3)
(4)
(8)
(9)
(10)
(12)
(13)
(a) FS with M=2, N=2 (M=N)
(b) FS with M=2, N=3 (M<N)
(c) FS with M=3, N=2 (M>N)
Collided
Successful
Idle
(*) T_c
Solved by the commands from the peer node
(It will not occupy a dedicated time slot during identification)
Depth is defined as the number of braches from that node to the root (top) node

Fig. 1. The FS mechanism with different M and N

III. The Proposed FSA-CSS Algorithm

A. Fast splitting mechanism

In the fast splitting (FS) mechanism, the counter value T_c of all tags are initialized to zero. When a collision occurs, the collided tags (i.e., tags with $T_c = 0$) perform $T_c = T_c + T_R$, while the tags with $T_c{>}0$ perform $T_c = M \cdot T_c + T_R$. That is,

$$T_c = \begin{cases} T_c + T_R, & \text{if } T_c = 0 \\ M \cdot T_c + T_R, & \text{if } T_c > 0\,. \end{cases} \tag{1}$$

where $T_R \in \{0,\ 1,\ \cdots,\ N-1\}$ denotes a random binary number generator at an arbitrary time, N determines the maximum subsets needed. That is to say, the collided tags can be potentially allocated into N subsets. T_c denotes the value of counter at an arbitrary time. An integer $M{>}0$ is a splitting coefficient, which denotes the splitting level of collided tags. The larger the M, the more available subset slots can be provided during a single splitting process.

In order to illustrate the FS mechanism, we describe a simple case with five tags using $M = N = 2$. Assume the tags a, b, c, d, e collide in the slot 0, given $T_c = 0$ is for all tags, the reader will randomly divide them into two subsets, e.g., $S_1 = \{a, d, e\}$ with random number 0 and $S_2 = \{b, c\}$ with random number 1. The FS mechanism will trigger the reader to split S_2 into $\{b\}$ and $\{c\}$ in slot 1. According to (1), no collision will happen between b and c. Meanwhile, tags a, d, and e will collide again in slot 1 and the FS will continue to assist the splitting until all collisions are solved. It is noted that we consider the case where a tag can be identified only once, whereas tags are identified repeatedly can be referred to [9][20-21].

By generalizing cases with $N{>}2$, we can derive the following result for the splitting coefficient M.

Result 1. *The performance of FS mechanism depends on the choice of M and leads to the following result.*

$$\begin{cases} M > N, & \textit{yield extra idle slots} \\ M = N, & \textit{no extra idle and collision slots} \\ M < N, & \textit{yield extra collision slots} \end{cases} \tag{2}$$

Proof: See the Appendix A. □

According to the Result 1, $M = N$ is the best choice for FS mechanism. Examples of four tags using FS mechanism with different M and N are illustrated in Fig. 1. The depth of a node is defined as the path length (number of solid branches) from that node to the root (top) node. As can be seen in Fig. 1 (a), all tags collided in slot 0 are further divided into two subsets, e.g., $S_1 = \{a, c\}$, $S_2 = \{b, d\}$. Since tags in S_1 are continually collided in slot 1, they act $T_c = T_c + T_R$, while the tags in S_2 act $T_c = 2 \cdot T_c + T_R$. After reading in slot 1, all tags are divided into four subsets. Each subset contains only one tag. In total, the FSA-CCS consumes six slots to identify four tags by using FS mechanism with $M = N = 2$. Fig. 1 (b) shows the example of $M{<}N$. In the depth 1, tag a acts $T_c = T_c + T_R = 0 + 2 = 2$, while tag d also acts $T_c = 2 \cdot T_c + T_R = 2 + 0 = 2$. As a result, two tags from different sets generate the same T_c, which results in an extra collision slot. Similarly as can be observed in Fig. 1 (c), many idle slots are introduced due to $M{>}N$. It thus indicates that

TABLE I
THE CHARACTERISTICS OF DIFFERENT ANTI-COLLISION ALGORITHMS

Method	Category			implementation	tag starvation	cardinality sensitive	system throughput	identification lantency	complexity
	DFSA	BS	Hybrid						
MAP [3]	$\surd$			medium	$\surd$	$\surd$	medium	medium	high
MFML-DFSA [4]	$\surd$			medium	$\surd$	$\surd$	medium	medium	high
FuzzyQ [5]	$\surd$			medium	$\surd$	$\surd$	low	medium	medium
ILCM [15]	$\surd$			easy	$\surd$	$\surd$	low	low	low
ds-DFSA [13]	$\surd$			difficult	$\surd$	$\surd$	high	low	low
BS [10]		$\surd$		easy	$\times$	$\times$	medium	high	low
ABS [9]		$\surd$		easy	$\times$	$\times$	medium	high	low
ABTSA [23]			$\surd$	difficult	$\surd$	$\surd$	high	high	high

a large M may cause too many idle slots. In our proposed algorithm, we define such a result with a large number of idle slots as over-splitting.

B. The proposed FSA-CSS algorithm

According to the analysis above, the FS mechanism can accelerate the splitting speed and reduce collision during the identification process. However, the question is how to apply the FS mechanism and avoid over-splitting, given the number of tags is unknown at the beginning? To tackle the over-splitting problem, we propose the fast splitting algorithm with consecutive slot status detection (FSA-CSS). The main idea is that the reader implements the FS mechanism only when it detects a consecutive collision. The consecutive collision indicates a large number of tags coexist in the same identification process. In contrast, when the reader detects a consecutive idle, it performs the shrink mechanism by sending a **Decrease command** to decrease counter value of tags. The shrink mechanism is defined as follows.

$$T_c = \text{trunk}\left(T_c/2\right) . \tag{3}$$

where trunk() is a truncation function. The flowchart of the proposed FSA-CSS algorithm is illustrated in Fig. 2. Two thresholds μ and σ are denoted as the upper tolerance limit of the number of consecutive collision slots and idle slots, respectively, and two counters T_{CCN} and T_{CIN} are used at the reader to count the number of consecutive collision slots and idle slots. If $T_{CCN} \geq \mu$, the reader should perform the FS mechanism. If $T_{CIN} \geq \sigma$, the reader should perform the shrink mechanism due to the occurrence of over-splitting.

As can be observed in Fig. 2, the reader sends the corresponding feedback commands according to the slot type. Similar to BS, the reader also has a counter R_c to terminate the identification process when $R_c < 0$. By receiving feedback commands from the reader, each tag acts as follows.

- **Splitting 0 command**: Tags with $T_c = 0$ act $T_c = T_R$, while tags with T_c>0 act $T_c = T_c + 1$. It is similar to the `ID-collision` feedback of the BS algorithm. After sending this command, the reader acts R_c=R_c+1.
- **Splitting 1 command**: Tags with $T_c = 1$ act $T_c = T_R$. Tags with T_c>0 will be split into two groups. The potential advantage is to reduce the collisions.
- **Increase command**: All tags act $T_c = 2 \cdot T_c + T_R$. Beside the current collided tags, tags with T_c>0 will be divided after receiving this command. This is essentially different to the BS algorithm. After sending this command, the reader acts R_c=2$\cdot$ R_c+1.
- **Decrease command**: All tags act T_c = trunk$(T_c/2)$. This command is allowed to alleviate the over-splitting. After sending this command, the reader acts R_c = round$(R_c/2)$.
- **QueryRep command**: All tags act $T_c = T_c - 1$. This is similar to the `ID-idle` and `ID-success` of the BS algorithm. After sending this command, the reader acts R_c=R_c-1.

It is worth noting that the optimal M cannot be directly derived by the algorithm. In this paper, we choose an appropriate value M by using experimental method. Intuitively, a larger M can provide more splitting subsets and hence reduce collisions. However, it also introduces idle slots. In our proposed solution, we choose $M = N = 2$ to best achieve the simplicity and balance between the performance and complexity of using random number generator by each tag. Additional analysis and discussion can be found from Section IV and Fig. 3 which compares the number of slots consumed by FSA-CSS by adopting different M.

Fig. 4 illustrates an example of seven tags by implementing the proposed FSA-CSS algorithm. Assume $\mu = 3$ and $\sigma = 3$, the reader performs the FS mechanism in the slot 3 due to $T_{CCN} = 3$. Then the reader performs the BS in each tag subset. After the tag c is identified, three consecutive idle slots occur, the reader performs the shrink mechanism in slot 11 to decrease the counter value of unidentified tags in order to alleviate over-splitting. Thereafter, the tags g, e and f are identified in slot 13, slot 15 and slot 16, respectively. The detailed communication procedure is also illustrated in Tab. II. In essence, compared to the conventional BS algorithm, the proposed FSA-CSS algorithm can reduce collisions but introduce extra idle slots. However, the FSA-CSS can use the **Increase command** to alleviate the negative impact from the idle slots.

C. Upper bound performance of FSA-CSS

In this section, we analyze the system throughput of FSA-CSS. The system throughput T_{sys} is defined as follows

$$T_{sys} = \frac{m}{N_m} . \tag{4}$$

where m is the number of tags waiting to be identified in the reader vicinity, N_m is the required total slots to identify the m tags. The system throughput is actually equivalent to the

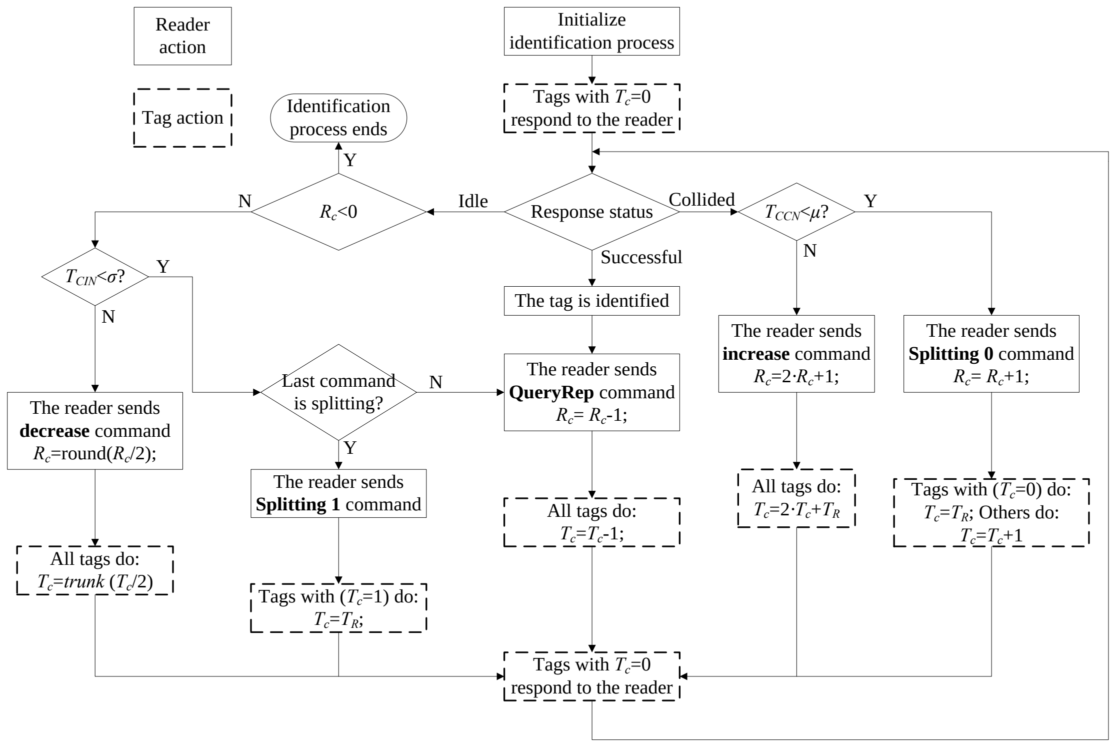
Reader action
Tag action
Initialize identification process
Tags with T_c=0 respond to the reader
Identification process ends
Y
N
R_c<0
Idle
Response status
Collided
T_{CCN}<μ?
Y
N
Successful
T_{CIN}<σ?
Y
N
The tag is identified
The reader sends increase command R_c=2·R_c+1;
The reader sends Splitting 0 command R_c= R_c+1;
Last command is splitting?
N
The reader sends QueryRep command R_c= R_c-1;
The reader sends decrease command R_c=round(R_c/2);
Y
The reader sends Splitting 1 command
All tags do: T_c=2·T_c+T_R
Tags with (T_c=0) do: T_c=T_R; Others do: T_c=T_c+1
All tags do: T_c=T_c-1;
All tags do: T_c=trunk (T_c/2)
Tags with (T_c=1) do: T_c=T_R;
Tags with T_c=0 respond to the reader

Fig. 2. The flowchart of the proposed FSA-CSS

TABLE II
AN EXAMPLE OF IDENTIFICATION PROCESS USING FSA-CSS

slot	Tag A (T_c, T_R)	Tag B (T_c, T_R)	Tag C (T_c, T_R)	Tag D (T_c, T_R)	Tag E (T_c, T_R)	Tag F (T_c, T_R)	Tag G (T_c, T_R)	Action	feedback
1	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(0, 1)	(0, 1)	(0, 1)	Collided	Splitting 0
2	(0, 0)	(0, 0)	(0, 0)	(0, 0)	(1, x)	(1, x)	(1, x)	Collided	Splitting 0
3	(0, 0)	(0, 0)	(0, 1)	(0, 1)	(2, 1)	(2, 1)	(2, 0)	Collided	Increase
4	(0, 1)	(0, 0)	(1, 1)	(1, 0)	(5, 1)	(5, 0)	(4, 1)	Collided	Increase
5	(1, x)	(0, x)	(3, x)	(2, x)	(11, x)	(10, x)	(9, x)	B is identified	QueryRep
6	(0, x)	-	(2, x)	(1, x)	(10, x)	(9, x)	(8, x)	A is identified	QueryRep
7	-	-	(1, x)	(0, x)	(9, x)	(8, x)	(7, x)	D is identified	QueryRep
8	-	-	(0, x)	-	(8, x)	(7, x)	(6, x)	C is identified	QueryRep
9	-	-	-	-	(7, x)	(6, x)	(5, x)	Idle	QueryRep
10	-	-	-	-	(6, x)	(5, x)	(4, x)	Idle	QueryRep
11	-	-	-	-	(5, x)	(4, x)	(3, x)	Idle	Decrease
12	-	-	-	-	(2, x)	(2, x)	(1, x)	Idle	Decrease
13	-	-	-	-	(1, x)	(1, x)	(0, x)	G is identified	QueryRep
14	-	-	-	-	(0, 0)	(0, 1)	-	Collided	Splitting 0
15	-	-	-	-	(0, x)	(1, x)	-	E is identified	QueryRep
16	-	-	-	-	-	(0, x)	-	F is identified	End identification

average number of identified tags per slot. In the FSA-CSS, the reader splits the collided tags via the fast splitting mechanism with $M = N = 2$. The identification process can be viewed as a binary tree. The depth of a node is the path length from that node to the root (top) of the tree. Assuming the FSA-CSS identifies m tags using the maximum L-th depth of the binary tree, the identification process can be viewed as a 2^L space to accommodate the m units. If the generated random number of all tags are uniformly distributed, the probability that r tags involved in a node on the L-th depth of the tree is

$$P\left(r|m,\ L\right) = C_m^r\left(\frac{1}{2^L}\right)^r \cdot \left(1-\frac{1}{2^L}\right)^{m-r}. \tag{5}$$

We then obtain the following expression for the probabilities of idle, success, and collision results from an independent probe at L-th depth of the tree.

$$P\left(0|m,\ L\right) = \left(1-\frac{1}{2^L}\right)^m, \tag{6}$$

$$P\left(1|m,\ L\right) = \frac{m}{2^L}\left(1-\frac{1}{2^L}\right)^{m-1}, \tag{7}$$

$$\begin{aligned} P\left(r>1|m,\ L\right) &= 1-P\left(0|m,\ L\right)-P\left(1|m,\ L\right) \\ &= 1-\left(1-\tfrac{1}{2^L}\right)^m-\tfrac{m}{2^L}\left(1-\tfrac{1}{2^L}\right)^{m-1}, \end{aligned} \tag{8}$$

As can be observed from Fig. 4, the reader always performs the BS to identify tags after the FS mechanism is finished. We

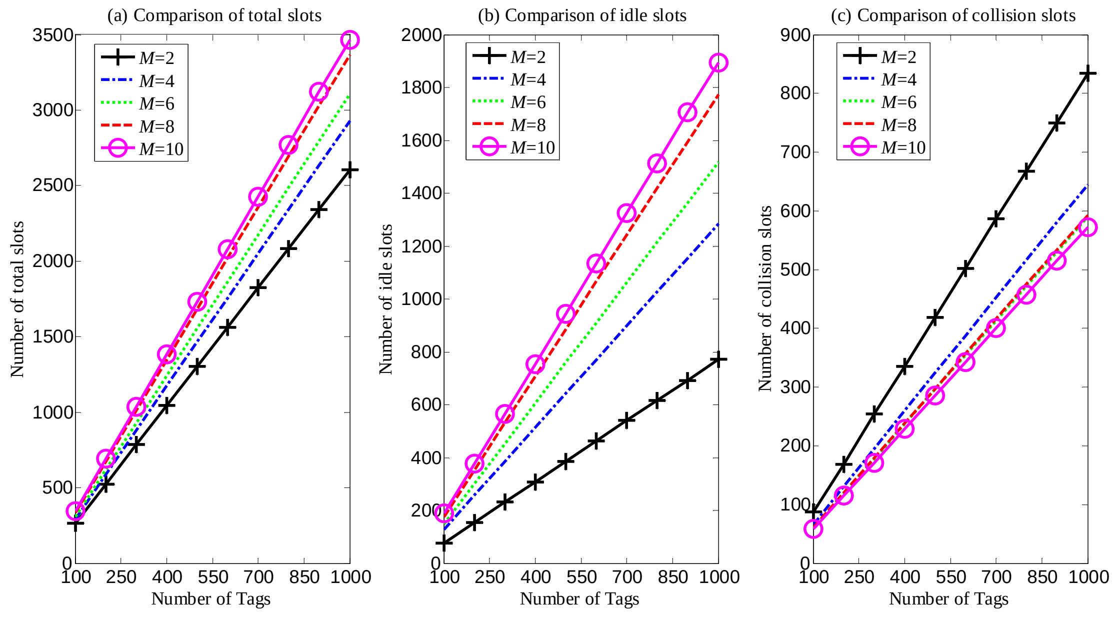

Fig. 3. Comparison of the number of slots under different M

assume that the reader does not perform the FS after the L-th depth, which is a reasonable assumption since the tags will be divided into 2^L subsets at L-th depth of the tree and each subset only contains a few tags, which reduces the chances of consecutive collisions. Thus, the total slots to identify m tags expended by FSA-CSS can be expressed as

$$\begin{aligned} N_m^{FSA-CSS} &= L + 2^L \cdot \left[P\left(0|m,\ L\right) + P\left(1|m,\ L\right)\right] \\ &+ \sum\nolimits_{r>1} 2^L \cdot P\left(r|m,\ L\right) \cdot N_r^{BS} \,. \end{aligned} \tag{9}$$

where N_r^{BS} denotes the number of slots used for BS to identify r tags. L represents the number of collided slots and involves $L - \mu$ times FS operations. When the FS is finished, the collided tags can be divided into 2^L groups. $2^L \cdot P\left(0|m,\ L\right)$ and $2^L \cdot P\left(1|m,\ L\right)$ represent the number of idle nodes and success nodes, respectively.

Result 2. *The upper bound system throughput of FSA-CSS can be approximated as following when the number of tags m tends to infinity*

$$\lim_{m\to\infty} T_{sys}^{FSA-CSS} \approx 0.4521\,. \tag{10}$$

Proof: See the Appendix B. □

D. Discussion of μ and σ

It is noted that Result 2 reveals the upper performance of FSA-CSS. However, the priori knowledge of number of tags is usually unknown to the reader and the frame size is typically not equal to the number of tags. In such cases, the FS and shrink mechanisms are necessary for the FSA-CSS to reduce collisions and avoid over-splitting during the entire identification process. Specifically, μ is the critical value to enable the FS mechanism, σ is the key value to perform the shrink mechanism. The system throughput depends on the setting of these two parameters.

We use the numerical method to search the optimal combination of μ and σ. Experiments are performed under the following cases: the number of tags are set as 200, 400, 600, and 800, respectively. μ and σ vary from 1 to 10 in steps of 1. As can be observed from Fig. 5, the optimal system throughput relies on the combination of μ and σ. In Fig. 5 (a), the maximum system throughput of FSA-CSS is 0.4049 which can be obtained when $\mu = 4$ and $\sigma = 9$. Similarly in Fig. 5 (b), the system throughput of FSA-CSS peaks at 0.4128 when $\mu = 4$ and $\sigma = 9$. Fig. 5 (c) shows that the pair of $\mu = 4$ and $\sigma = 7$ can achieve the maximum system throughput of 0.4178. In Fig. 5(d), $\mu = 4$ and $\sigma = 7$ can also achieve the best performance at 0.4188 when $n = 800$. In general, no constant parameter setting of μ and σ can maintain the best performance as the number of tags varies in a large scale. The optimal values of μ and σ rely on exhaustive search using computer simulations.

The system throughput of FSA-CSS under specific pairs of μ and σ when the number of tags varies from 100 to 1000 are illustrated in Fig. 6. The average throughput of six curves from highest to lowest are 0.4128, 0.4123, 0.4122, 0.4102, 0.4086, and 0.4083, respectively. The corresponding pairs of μ and σ are (4, 7), (4, 8), (4, 6), (4, 9), (3, 4), and (3, 5), respectively. We can found that no constant parameter setting can always maintain the best performance. When the number of tags varies in a large-scale, the expectation of system throughput of FSA-CSS is maximum at ($\mu = 4$, $\sigma = 7$). In practical implementation of fast RFID systems, it might be too costly to search the optimal combination of these two parameters and keep real-time updating by introducing extra overhead strategy. Therefore, a default parameter setting is preferable during the whole identification process with recommendation of ($\mu = 4$, $\sigma = 7$).

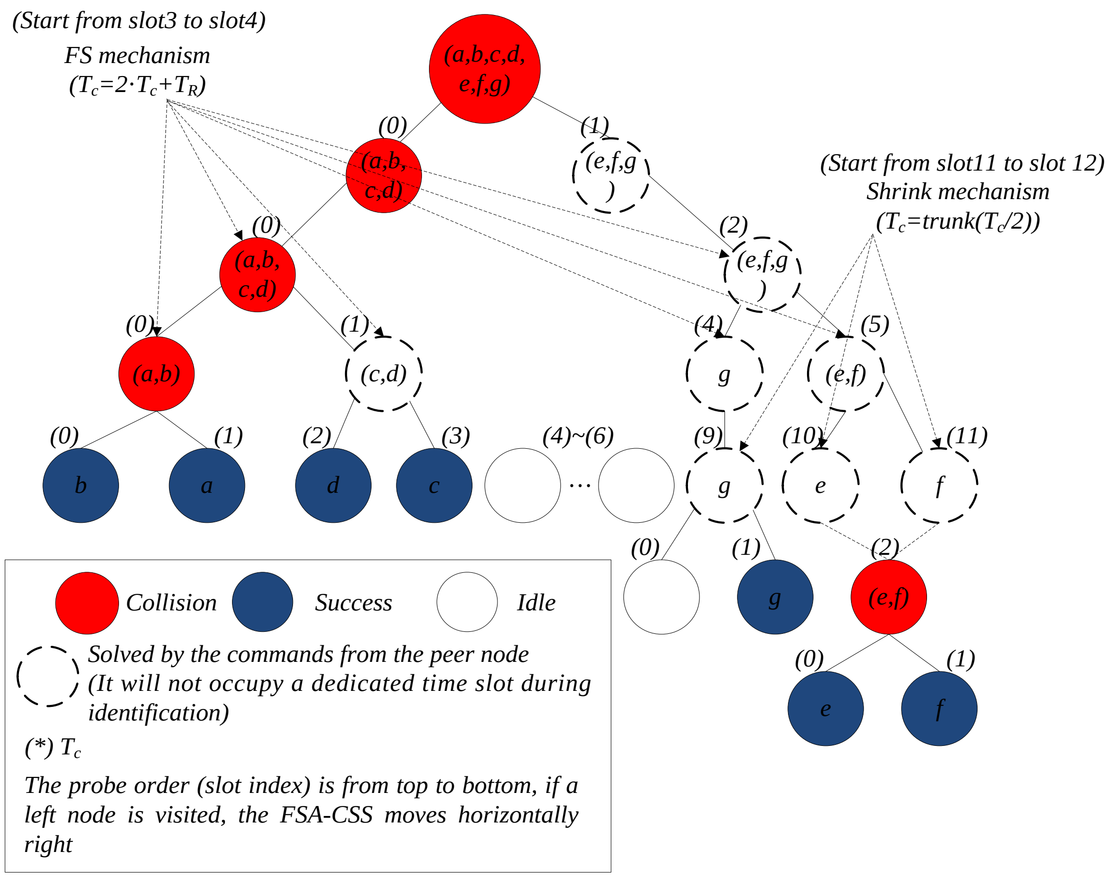
(Start from slot3 to slot4)
FS mechanism
(T_c=2·T_c+T_R)
(a,b,c,d,e,f,g)
(0)
(1)
(a,b,c,d)
(e,f,g)
(Start from slot11 to slot 12)
Shrink mechanism
(T_c=trunk(T_c/2))
(2)
(e,f,g)
(0)
(a,b,c,d)
(0)
(1)
(4)
(5)
(a,b)
(c,d)
g
(e,f)
(0)
(1)
(2)
(3)
(4)~(6)
(9)
(10)
(11)
b
a
d
c
g
e
f
(0)
(1)
(2)
g
(e,f)
(0)
(1)
e
f
Collision
Success
Idle
Solved by the commands from the peer node (It will not occupy a dedicated time slot during identification)
(*) T_c
The probe order (slot index) is from top to bottom, if a left node is visited, the FSA-CSS moves horizontally right

Fig. 4. An identification example by using FSA-CSS algorithm

IV. Simulation Results

In this section, we compare FSA-CSS with the existing state-of-the-art including ABS [9], APAFQ [17], ds-DFSA [13], EAAEA [16], ILCM [15], and ABTSA [23] in terms of system throughput and time efficiency. Simulations with a reader and a various number of tags have been evaluated using MATLAB, where the tags are uniformly distributed in the reader vicinity in order to receive reader command directly. The evaluations are mainly focused on the MAC layer, whereas physical layer effects such as radio propagation effects are not considered in the proposed model. The similar assumption has been widely applied in the literatures [4, 10, 13-16]. The simulation results are average over 1000 iterations.

Fig. 7 illustrates the simulation results of system throughput under different initial frame size. Since ABS and FSA-CSS are not Aloha-based algorithms, their performances are not affected by varying initial frame size. The FSA-CSS only shows minor improvement when the number of tags is below 200. The reason is that the frequent use of FS mechanism introduces too many idle slots when the size of tag is small, and hence decreases the system throughput. As the number of tags is increased, the FSA-CSS shows its advantage over other algorithms. From Fig. 7, the FSA-CSS outperforms all other algorithms and achieves an average system throughput of 0.4128, where the average throughput of ABS, APAFQ, ABTSA, ds-DFSA, EAAEA, and ILCM are 0.3448, 0.3573, 0.4083, 0.4079, 0.3361, and 0.3252, respectively. It is noted that although ABTSA adopts a complex hybrid structure to enhance its performance, our proposed FSA-CSS still outperforms ABTSA by 1.10%. Similarly, the FSA-CSS is superior to the ds-DFSA and APAFQ with lower implementation cost.

In Fig. 8, we compare FSA-CSS ($\mu = 4$, $\sigma = 7$) with ABS in terms of total number of slots, collision slots and idle slots. The number of unread tags is progressively increased from 100 to 1000. Compared to ABS, FSA-CSS has fewer collision slots but more idle slots. The simulation results indicate that significant collision slots are reduced by the FS mechanism, and hence the total slots consumed by the FSA-CSS. Although the shrink mechanism is expected to overcome the over-splitting, the extra idle slots are unavoidable by the FS mechanism. As a result, the practical performance of FSA-CSS is slightly lower than its theoretical results.

Consider the disparity between slot durations such as the duration of non-idle slot is always longer than that of idle slot [22], the system throughput metric is ineffective to evaluate the performance of identification in terms of identification time. Therefore, we use time efficiency in the simulations. Assume the time durations of success, collision and idle slots are denoted as T_s, T_c and T_e, the time efficiency is defined as [16]

$$T_{effi} = \frac{S \cdot T_s}{S \cdot T_s + E \cdot T_e + C \cdot T_c}. \tag{11}$$

where E, S, and C are the statistics of success, idle and collision during the identification process measured by the

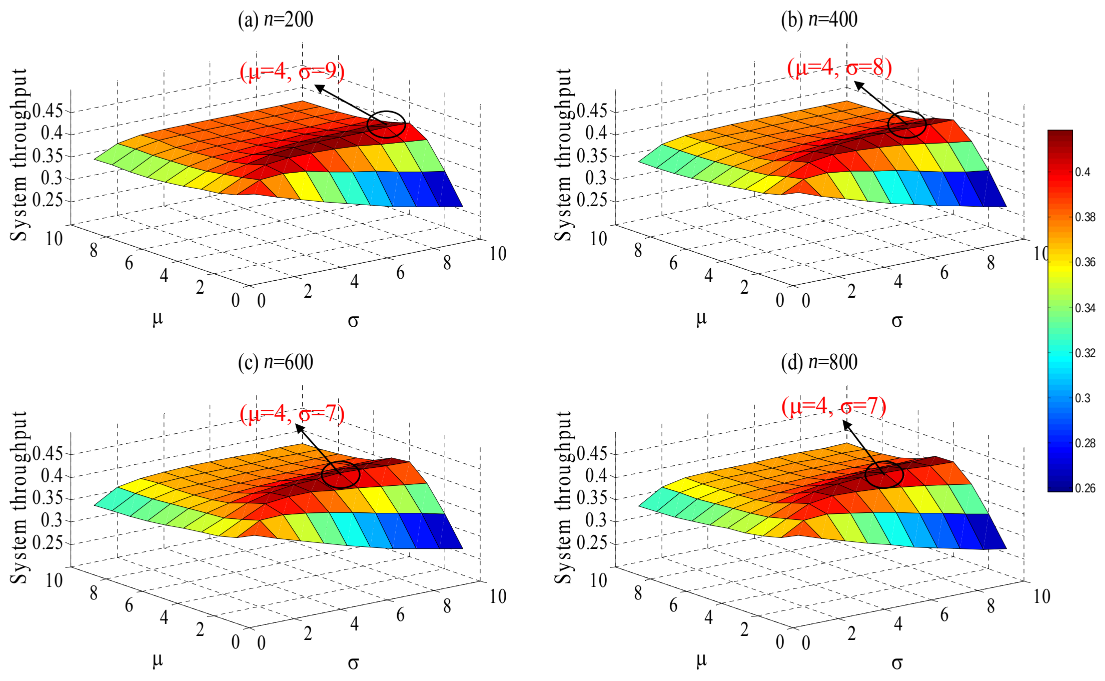

Fig. 5. Comparison of system throughput under different number of tags

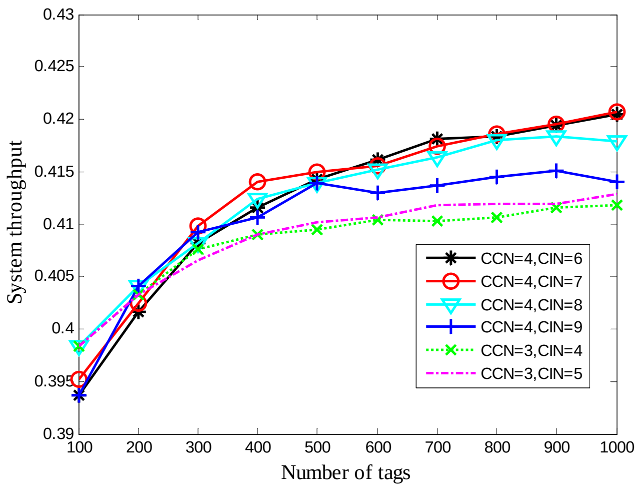

Fig. 6. The system throughput of FSA-CSS under specific pairs of μ and σ

reader.

Fig. 9 shows the time efficiency of various methods under different ratios between the duration of success, collision and idle slots. As can be seen, all algorithms present fluctuating performances under different $T_s : T_c : T_e$ ratios. When $T_c = T_e$ in Fig. 9 (a) and (b), all the algorithms present a similar behavior, and the performance ranking from high to low is FSA-CSS, ds-DFSA, ABTSA, APAFQ, ABS, EAAEA, and ILCM. Fig. 9 (c) and (d) show differently with the ranking of ABS dropping down to the last one and the ranking of APAFQ moving up to the third when $T_s : T_c : T_e = 3 : 3 : 1$. As the ratio between T_c and T_e is increased, ABS shows a significant performance degradation due to the large number of collision slots. Compared to the reference methods, the proposed FSA-CSS can always achieve the best time efficiency under various ratios by reducing collision and idle slots.

V. Experimental Results With A Practical RFID Testbed

TABLE III
Link Parameters Used For RF Communications

Parameter	Value
Frequency	922.875 MHz
Backscatter link frequency	64 KHz
Modulation	DSB-ASK
Deviation	20 HHz
Channel width	250 KHz
Read-tag data coding	TPP
Tag-reader data coding	Miller-8
Reader-tag preamble	137.5 μs
Tag-reader preamble	2000 μs
T1	154 μs
T2	130 μs
T3	436 μs

To further evaluate the performance of FSA-CSS algorithm in a practical UHF RFID system, we conduct experiments using a testbed in an indoor environment. Experiments include an active RFID reader and 20 passive tags. The reader is equipped with ARM Cortex A9 processor, which is a 32-bit reduced instruction set (RISC) processor with a maximum operating frequency of 1 GHz and an off-chip memory 512M to ensure high speed and stable operation of programs. The external interface of the reader includes UART, JTAG, ETH and USB, which greatly facilitates the software and hardware debugging. The UART interface is used for communication between the host computer and the reader. The JTAG interface facilitates the developer to debug the reader hardware. ETH

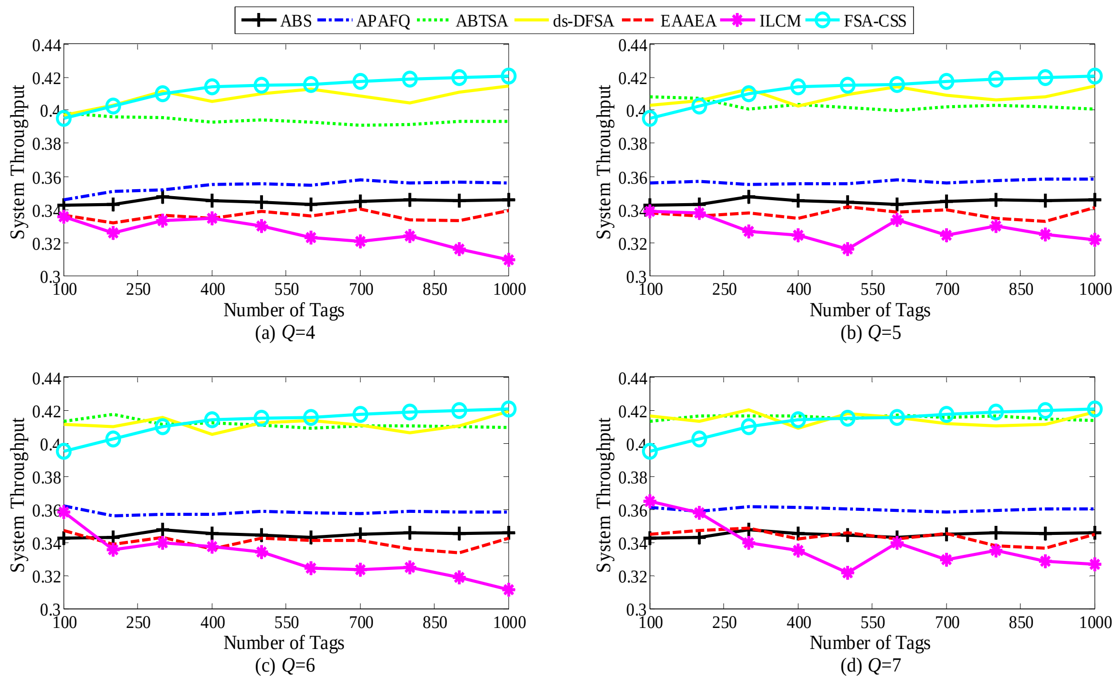

Fig. 7. System throughput comparisons of various algorithms under different initial frame size

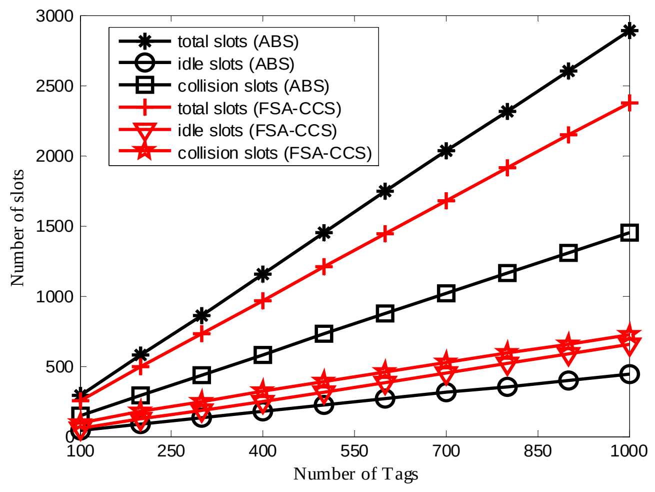

Fig. 8. Simulation results of the number of total slots, collision slots and idle slots between FSA-CSS and ABS

and USB provide various ways to upload tag data by the reader. Compared to the BS commercial tag, the custom tag has added some control logic codes in its state machine, which is used to support the reader commands of FSA-CSS.

Tab. III lists the link parameters configured for radio frequency communication between the reader and tags. In order to comply with ISO/IEC 18000-6B, the carrier frequency is set to 922.875MHz and Miller coding is used.

The experiments are carried out by placing 20 tags in the antenna interrogation zone of RFID reader with a fixed transmitting power. We evaluate and compare the performance of standard BS used in ISO/IEC 18000-6B and the proposed FSA-CSS in total identification time (defined as the total time required to identify all tags) and identification rate (defined as the number of tags can be identified per second). The experimental environment is captured in Fig. 10. We vary the number of tags from 2 to 20 and repeat the experiment in 50 trials to get average performance. In parallel, we also perform simulations using the same parameters.

Fig. 11 compares the experimental results of FSA-CSS with that of ABS which is used in ISO/IEC 18000-6B. In the experiments, the proposed FSA-CSS reduces the identification time by 32.5% and improves the average identification rate by 50% compared with ABS. The simulation results are closed to the experimental results. Both simulation and experiment results indicate that the proposed FSA-CSS outperforms the ABS constantly in the practical RFID system.

VI. Conclusions

We have proposed a novel design of BS-based anti-collision algorithm namely FSA-CSS to improve the low identification efficiency of traditional BS-based algorithms. Unlike the existing methods, the FSA-CSS allows tags which are in the waiting state to participate in the splitting process via fast splitting mechanism, thus the number of collision slots can be reduced. Meanwhile, the shrink mechanism has been introduced to avoid the over-splitting problem. The proposed FSA-CSS has been shown to improve the identification efficiency without estimation of number of tags and extra hardware cost. The simulation results have shown that FSA-CSS outperforms ABS by 23.3% in system throughput. We have also prototyped a RFID system and evaluated its performance in a real-world RFID environment. The experimental results have been shown that the proposed FSA-CSS can reduce the total identification time by 32.5% for identifying 20 tags. Both simulations and

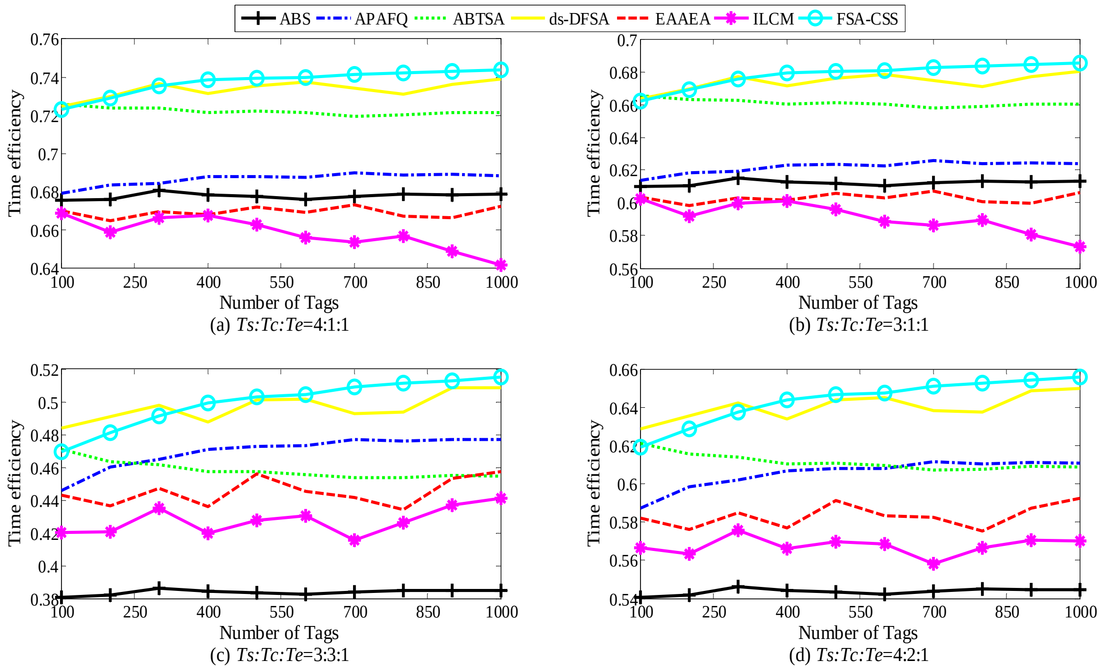

Fig. 9. Simulation results of time efficiency for various algorithms under varying $T_s : T_c : T_e$

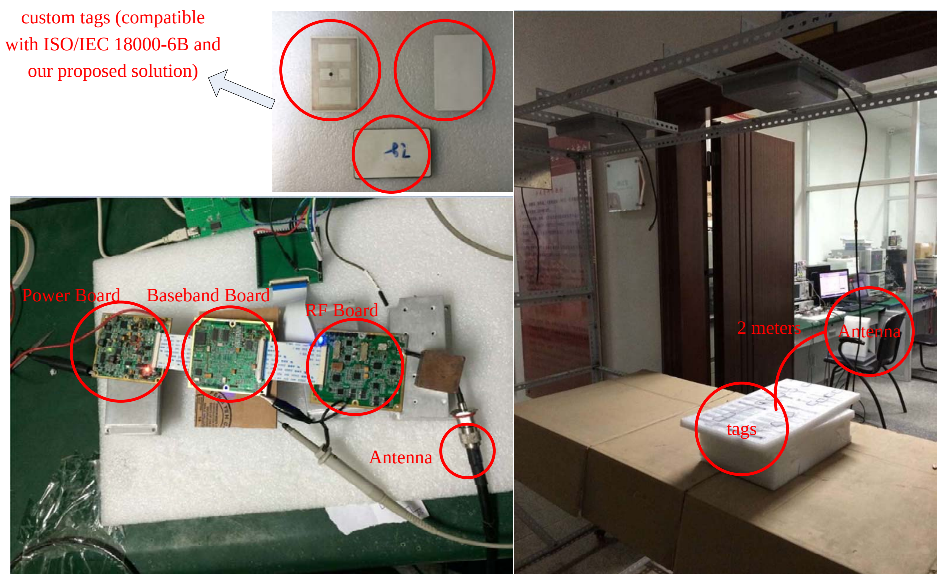
custom tags (compatible with ISO/IEC 18000-6B and our proposed solution)
Power Board
Baseband Board
RF Board
Antenna
2 meters
Antenna
tags

Fig. 10. The hardware setup used in the experiments

experiments have indicated that the proposed FSA-CSS is a suitable candidate for the resource-constrained RFID systems.

Appendix A
Proof of Result 1

Assuming the i-th and j-th ($i<j$) subsets on the depth k have counters $T_c^{(i,\,k)}$, $T_c^{(j,\,k)}$, respectively, we have

$$T_c^{(j,\,k)} = T_c^{(i,\,k)} + (j-i)\,, \tag{12}$$

If the FS mechanism is performed when a collision occurs, the two subsets extend into the depth $(k+1)$, and the counters can be expressed as

$$T_c^{(i,\,k+1)} = M \cdot T_c^{(i,\,k)} + T_R^{(i,\,k)}\,, \tag{13}$$

$$T_c^{(j,\,k+1)} = M \cdot T_c^{(j,\,k)} + T_R^{(j,\,k)}\,, \tag{14}$$

where $T_R^{(i,\,k)}$, $T_R^{(j,\,k)} \in \{0,\ 1,\ \cdots,\ N-1\}$. According to Eqs. (3)-(5), $T_c^{(j,\,k+1)}$ can be rewritten as

$$T_c^{(j,\,k+1)} = M \cdot \left[T_c^{(i,\,k)} + (j-i)\right] + T_R^{(j,\,k)}\,. \tag{15}$$

Comparing between Eq. (13) and (15), we have

$$T_c^{(j,\,k+1)} - T_c^{(i,\,k+1)} = M \cdot (j-i) + \left(T_R^{(j,\,k)} - T_R^{(i,\,k)}\right)\,, \tag{16}$$

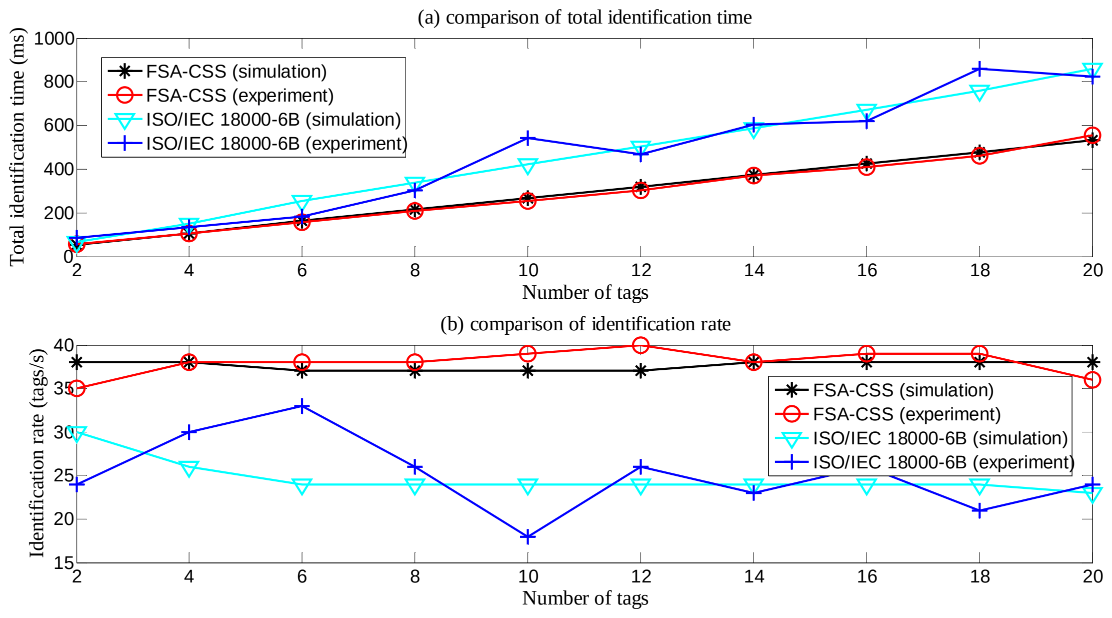

Fig. 11. Comparison of the experimental results

Since $T_R^{(i,\,k)}$, $T_R^{(j,\,k)} \in \{0,\ 1,\ \cdots,\ N-1\}$, we have

$$-(N-1) \le T_R^{(j,\,k)} - T_R^{(i,\,k)} \le (N-1)\ , \tag{17}$$

Without loss of generality, we assume $0<M<N$, then we have

$$0 < M \cdot (j-i) < N(j-i)\,, \tag{18}$$

Since $j>i$, (18) can be rewritten as

$$M < M \cdot (j-i) < N(j-i)\,, \tag{19}$$

According to (16), (17), and (19), we have

$$M-(N-1) \le T_c^{(j,\,k+1)} - T_c^{(i,\,k+1)} \le N(j+1-i)-1\,, \tag{20}$$

Since i, j, M and N are natural number above 0, (20) can be re-expressed as

$$0 \le T_c^{(j,\,k+1)} - T_c^{(i,\,k+1)} \le N(j+1-i)-1\,. \tag{21}$$

which indicates that the nodes in different subsets at the depth k may be collided in the same subset at depth $(k+1)$. Consequently, the extra collision slot is yielded. Suppose $M=N$, we have

$$1 \le T_c^{(j,\,k+1)} - T_c^{(i,\,k+1)} \le N(j+1-i)-1\,. \tag{22}$$

which indicates that the nodes in two subsets at depth k extending into depth $(k+1)$ will not be collided. Based on the discussion above, we know that $M=N=2$ can avoid the extra collision slot.

On the other hand, if $M>N$, according to (20), we have

$$2 \le T_c^{(j,\,k+1)} - T_c^{(i,\,k+1)} \le N(j+1-i)-1\,. \tag{23}$$

which indicates that the minimum gap between counter value in adjacent subsets is above 1. That means the extra idle slot is generated when $M>N$ compared to (22). Therefore, the Result 1 can be yielded.

Appendix B
Proof of Result 2

As the prior knowledge of number of tags is known to the reader, it is straightforward that the FS should be repeated until all tags have been divided into m subsets and each subset contains few tags in equal probability. Then the conventional BS can be adopted to identify these tags in each subsets. The maximum depth L_{max} of the binary tree using FSA-CSS to identify m tags can be derived as

$$L_{\max} = \lfloor \log_2^m \rfloor\ , \tag{24}$$

where $\lfloor\cdot\rfloor$ denotes the operation of round down to the nearest integer. Substituting Eq. (24) into Eq. (9) leads to

$$\begin{aligned} N_n^{FSA-CSS} &= L_{\max} + 2^{L_{\max}} \cdot (1-\tfrac{1}{2^{L_{\max}}})^m \\ &+ m \cdot (1-\tfrac{1}{2^{L_{\max}}})^{m-1} \\ &+ \sum_{r=2}^{m} 2^{L_{\max}} \cdot P\left(r|m,\ L\right) \cdot N_r^{BS}\ , \end{aligned} \tag{25}$$

When $2_{max}^L = m$, FSA-CSS algorithm consumes the least number of slots to identify m tags. When $2_{max}^L = m$ and $m \to \infty$, the Eq. (25) can be further expressed as

$$\begin{aligned} & N_n^{FSA-CSS}\Big|_{2^{L_{\max}}=m,\ m\to\infty} \\ &= L_{\max} + 2^{L_{\max}} \cdot (1-\tfrac{1}{2^{L_{\max}}})^m + m \cdot (1-\tfrac{1}{2^{L_{\max}}})^{m-1} \\ &+ \sum_{r=2}^{m} \tfrac{2^{L_{\max}} \cdot m!}{r!(m-r)!} \left(\tfrac{1}{2^{L_{\max}}}\right)^r \cdot \left(1-\tfrac{1}{2^{L_{\max}}}\right)^{m-r} \cdot N_r^{BS} \\ &\approx L_{\max} + \tfrac{2\times 2^{L_{\max}}}{e} + \sum_{r=2}^{m} \tfrac{2^{L_{\max}}(e-1)}{e} \cdot N_r^{BS} \\ &\approx 2.212m\,. \end{aligned} \tag{26}$$

According to Eqs. (4) and (26), the Result 2 can be yielded.

References

[1] J. Wang, H. Hassanieh, D. Katabi, P. Indyk, "Efficient and reliable low-power backscatter networks," *Proc. 2012 ACM SIGCOMM*, pp. 61-72, Jul. 2012.

[2] X. Liu, B. Xiao, S. Zhang, K. Bu, and A. Chan, “Step: A time-efficient tag searching protocol in large RFID systems,” *IEEE Trans. Comput.*, vol. 64, no. 11, pp. 3265-3277, 2015.

[3] W.-T. Chen, “An accurate tag estimate method for improving the performance of an RFID anticollision algorithm based on dynamic frame length ALOHA,” *IEEE Trans. Autom. Sci. Eng.*, vol. 6, no. 1, pp. 9-15, 2009.

[4] J. Vales-Alonso, V. Bueno-Delgado, E. Egea-Lopez, et al., “Multiframe maximum-likelihood tag estimation for RFID anticollision protocols,” *IEEE Trans. Ind. Informat.*, vol. 7, no. 3, pp. 487-496, 2011.

[5] L. Arjona, H. Landaluce, A. Perallos, and E. Onieva, “Fast fuzzy anticollision protocol for the RFID standard EPC Gen-2,” *Electron. Lett.*, vol. 52, no. 8, pp. 663-665, 2016.

[6] J. Shin, B. Jeon, and D. Yang, “Multiple RFID tags identification with M-ary query tree scheme,” *IEEE Commun. Lett.*, vol. 17, no. 3, pp. 604-607, 2013.

[7] X. Jia, Q. Feng, and L. Yu, “Stability analysis of an efficient anti-collision protocol for RFID tag identification,” *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 2285-2293, 2012.

[8] H. Landaluce, A. Perallos, E. Onieva, L. Arjona, and L. Bengtsson, “An energy and identification time decreasing procedure for memeoryless RFID tag anticollision protocols,” *IEEE Trans. Wireless Commun.*, vol. 15, no. 6, pp. 4234-4247, 2016.

[9] J. Myung, W. Lee, and J. Srivastava, “Adaptive binary splitting for efficient RFID tag anti-collision,” *IEEE Commun. Lett.*, vol. 10, no. 3, pp. 144-146, 2006.

[10] Y. Cui and Y. Zhao, “Performance evaluation of a multi-branch tree algorithm in RFID,” *IEEE Trans. Commun.*, vol. 58, no. 5, pp. 1356-1364, 2010.

[11] Y. H. Chen, S. J. Horng. R. S. Run, et al., “A novel anti-collision algorithm in RFID systems for identifying passive tags,” *IEEE Trans. Ind. Inform.*, vol. 6, no. 1, pp. 105-121, 2010.

[12] C. Angerer, R. Langwieser, and M. Rupp, “RFID reader receives for physical layer collision recovery,” *IEEE Trans. Commun.*, vol. 58, no. 12, pp. 3526-3537, 2010.

[13] J. Su, Z. Sheng, D. Hong, and G. Wen, “An effective breaking policy for dynamic framed slotted aloha in RFID,” *IEEE Commun. Lett.*, vol. 20, no. 4, pp. 692-695, 2016.

[14] Y. Lai, L. Hsiao, and B. Lin, “Optimal slot assignment for binary tracking tree protocol in RFID tag identification,” *IEEE/ACM Trans. Netw.*, vol. 23, no. 1, pp. 255-268, 2015.

[15] P. Solic, J. Radic, and N. Rozic, “Energy efficient tag estimation method for Aloha-based RFID systems,” *IEEE Sensors J.*, vol. 14, no. 10, pp. 3637-3647, 2014.

[16] W.-T. Chen, “Optimal frame length analysis and an efficient anticollision algorithm with early adjustment of frame length for RFID systems,” *IEEE Trans. Veh. Technol.*, vol. 65, no. 5, pp. 3342-3348, 2016.

[17] J. Su, X. Zhao, D. Hong, Z. Luo, and H. Chen, “Q-value fine-grained adjustment based RFID anti-collision algorithm,” *IEICE Trans. Commun.*, vol. E99-B, no. 7, pp. 1593-1598, 2016.

[18] C. Qian, Y. Liu, R. H. Ngan, and L. M. Ni, “ASAP: Scalable collision arbitration for large RFID systems,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 7, pp. 1277-1288, Jul. 2013.

[19] L. Zhang, J. Zhang, and X. Tang, “Assigned tree slotted Aloha RFID tag anti-collision protocols,” *IEEE Trans. Wireless Commun.*, vol. 12, no. 6, pp. 5493-5505, 2013.

[20] I. Bratuz, A. Vodopivec, and A. Trost, “Resolving collision in EPCglobal Class-1 Gen-2 system by utilizing the preamble,” *IEEE Trans. Wireless Commun.*, vol. 13, no. 10, pp. 5330-5339, 2014.

[21] Y. Lai and C. Lin, “”Two blocking algorithms on adaptive binary splitting: single and pair resolutions for RFID tag identification,” *IEEE Trans. Netw.*, vol. 17, no. 3, pp. 962-975, 2009.

[22] Y. M. Hu, I. C. Chang, and J. S. Li, “Hybrid blocking algorithm for identification of overlapping staying tags between multiple neighboring readers in RFID systems,” *IEEE Sensors J.*, vol. 15, no. 7, pp. 4076-4085, 2015.

[23] H. Wu, Y. Zeng, J. Feng, and Y. Gu, “Binary tree slotted Aloha for passive RFID tag anticollision,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 6, pp. 19-31, 2013.