An Adapted RFID Anti-collision Algorithm in a Dynamic Environment

Weiping Zhu*, Mingzhe Li*, Jiannong Cao†, and Xiaohui Cui*‡
*School of Computer Science, Wuhan University, P. R. China
†Department of Computing, The Hong Kong Polytechnic University, Hong Kong
Email: {wpzhu,mzli,xcui}@whu.edu.cn, csjcao@comp.polyu.edu.hk

***Abstract*—In recent decades, radio frequency identification (RFID) has been used in many applications in the world. Tag anti-collision is a fundamental technique for RFID, for solving the collisions when multiple RFID tags transmit their IDs to an RFID reader simultaneously. This technique is well investigated in the stationary environment, however, has some deficiencies when the number of tags in the interrogation region of the reader changes dramatically. This paper proposes a tag anti-collision algorithm called TAD to solve this problem. TAD can effectively and fast estimate the number of arriving and leaving tags, and automatically adapt to different changes of the tags using a hybrid method. The simulation results show that TAD significantly outperforms existing approaches in the situation with many leaving tags.**

***Index Terms*—RFID, anti-collision, dynamic environment, threshold estimation.**

I. Introduction

Radio frequency identification (RFID) is an automatic identification technology that uses radio communication to obtain the IDs of objects. RFID is widely used in many applications such as supply chain management, human surveillance, and food safety management. A typical RFID system consists of tags and readers. Each tag has a microchip which stores its ID and some other information. A reader transmits a query to the tags, and the tags respond over a shared wireless medium. Tag collisions happen when more than one tag transmit their information simultaneously, causing the reader unable to identify any tag. Therefore, efficient approaches are required to reduce such collisions. These approaches are called tag anti-collision algorithms in the RFID field.

Existing tag anti-collision algorithms including tree-based algorithms [1], [2], [3], [4], [5] and Aloha-based algorithms [6], [7], [8], [9], [10] are mostly designed for stationary scenarios where tags do not move. J. Myung *et al.* proposed the adaptive binary splitting protocol (ABS) and adaptive query splitting protocol (AQS) to improve performance in the dynamic environment [11] where tags may move in and out the interrogation region of the reader. In order to improve the identification efficiency of ABS, single resolution blocking ABS algorithm (SRB) and pair resolution blocking ABS algorithm (PRB) were proposed by distinguishing arriving tags from staying tags [12]. These approaches are further improved by slitting tags into more subsets or merging tags more fast (named FSA-CSS) [13], using collision bit detection and dual prefixes matching to reduce the number of reader queries and tag replies [14], transmitting fewer amount of data between the reader and tags [15]. These approaches show desirable performance in their target scenarios. However, their performance in the environment with dynamic changes of tags require further improved.

In this study, we propose a tag anti-collision algorithm called TAD for the highly dynamic environment where tags can move in and out the interrogation region of the reader frequently. In this algorithm, staying tags and arriving tags are distinguished and identified separately. In each identification, we estimate the number of staying tags and arriving tags, compute the staying ratio of tags, and then adaptively perform proper approaches to achieve better performance. The staying ratio estimation process is optimized in terms of execution time. In summary, this study makes the following contributions.

- We proposed a hybrid identification approach to adapt to the changes of staying ratio in the interrogation region of a reader.
- We proposed threshold estimation approach which is especially suitable to speed up the staying ratio estimation for the hybrid identification approach.
- We performed theoretical analysis and simulations for validating the performance of the proposed hybrid approach. The results show that the proposed approach achieves desirable performance compared with the existing approaches.

II. System Model

We assume that there is an RFID reader constantly identifying RFID tags in its interrogation region. The tags may move in and out. A typical identification process is that the reader sends a query to the tags, the tags reply their information, and the reader notifies them the results whether their information is received. This is called a time slot in the identification. A time slot is called an idle slot, readable slot, or collision slot when no tag responds, only one tag responds, or multiple tags respond, respectively [11]. A frame is defined as the time duration that a reader recognizes all tags in its interrogation region. Since the identification is performed repeatedly, there are multiple frames. Let f_i denote the ith frame. Some notations used in this study are listed as follows:

‡Corresponding Author. Email: xcui@whu.edu.cn

978-1-7281-3106-1/20/$31.00 ©2020 IEEE

- *Staying tags*: the tags that exist in both f_{i-1} and f_i.
- *Arriving tags*: the tags that do not exist in f_{i-1}, but exist in f_i.
- *Leaving tags*: the tags that exist in f_{i-1}, but do not exist in f_i.
- $|N_i|$: the number of tags existing in f_i.
- *Staying ratio*: the number of staying tags divided by $|N_{i-1}|$.
- *Arriving ratio*: the number of arriving tags divided by $|N_{i-1}|$.
- *preTagNum*: number of tags identified in the last frame.
- *arrEst*: estimated number of arriving tags .
- *stayingEst*: estimated number of staying tags .

In this study, we aim to achieve the minimum execution time for identifying all the tags, considering dynamic changes of tags in the reader's interrogation region.

III. Our Solution

In a dynamic environment, a tag anti-collision algorithm includes several parts, distinguishing staying tags and arriving tags, identifying arriving tags, and identifying staying tags. In this study, we follow ABS and its improvements for the identification. However, these approaches still have some problems. First, they do not adapt to different staying ratios in a dynamic environment. For example, PRB works well in a large staying ratio while ABS is suitable in a small staying ratio. Second, existing works such as PRB and ATQS [16] use the information of the last frame to estimate the number of arriving tags in the current frame, which is inaccurate if the tag arrivals change dramatically between two identifications. TAD solves the aforementioned problems.

A. Distinguish Arriving Tags from Staying Tags

Staying tags and arriving tags cannot completely be distinguished by using the ID of the reader. We adopt a technique similar with ECRB [17] to solve this problem. After identified, each tag stores the ID of the reader and the ID of next frame. At the beginning of each frame, the reader broadcasts its ID and the frame ID. On receiving these IDs, all tags compare them to their stored information correspondingly. If either of them are unequal, the tag is an arriving tag and set *isArr* to 1; otherwise it is a staying tag. Finally, all tags update their stored frame IDs to the ID of the next frame.

B. Tag Cardinality Estimation and Staying Ratio Estimation

We estimate the number of staying tags and arriving tags before identification. We use two different estimation approaches, exact estimation and threshold estimation, for arriving tags and staying tags, respectively. These approaches also can be used for staying ratio estimation as explained below.

First, we estimate the exact number of arriving tags. Lottery of frame (LoF) [18] approach is used for this purpose. The main idea is to request tags to reply following a geometric distribution and then estimate the number of tags based on the received replies. The reader sequentially broadcasts 1, 2, ... in time slots and listens to the replies from tags. A tag transmits a bit's information if the position of its ID's rightmost "1" equals to the received number and it is an arriving tag. For example, the tag with ID 11010 transmits its ID when receiving 2, because the rightmost "1" in its ID locates in the second from the right. The process is repeated until there is no tag replying. Then, the reader estimates the number of arriving tags as $1.2897 \times 2^{i-1}$ [18] and stores it in *arrEst*.

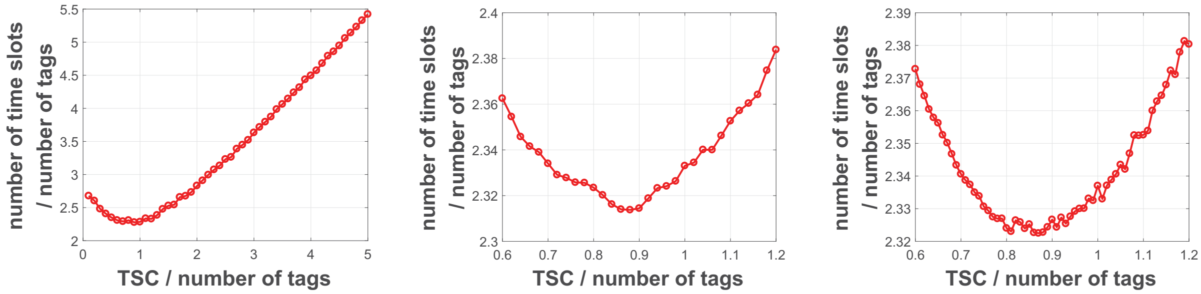
In some cases, there is no need to estimate the exact number of tags, but only determine whether the number of tags greater than a threshold. This is not only necessary for the anti-collision algorithms using rough estimation of tags, but also for the staying ratio estimation. Each staying ratio corresponds to a threshold of number of tags estimated. For example, in TAD, it is possible to terminate the staying tag estimation early before the first empty slot appears. The estimation of staying tags can be terminated when $i > \log_2 \frac{preTagNum \times 0.2155}{1.2897} + 1$ (the value will be explained in the next subsection), because under these circumstances the staying ratio is greater than the one for changing tag collision strategy.

C. Hybrid Identification Strategy

In a dynamic environment, the staying ratio of tags often changes and difficult for the anti-collision algorithms to always achieve desirable performance. We believe that a hybrid strategy is required to automatically adapt to the staying ratio. More specifically, different approaches are performed according to different staying ratios. In TAD, we demonstrate the hybrid strategy using PRB and ABS. PRB is used when the staying ratio is more than a threshold *changingRatio*, and otherwise ABS is used. It is noted that we have not restricted the candidate anti-collision approaches, and other more promising approaches can be used in a similar way.

We compute the *changingRatio* for TAD as follows. In ABS, it is known that the optimal number time slots (denoted by *TSC*) for *n* tags is 0.88*n* [17], [19]. It means that *n* tags are initialled their identifying sequences (denoted by *ASC*) between 0 to 0.88*n*). However, the total number of time slots required for *n* tags is not reported in the literature. We conducted simulations to determine this value for the first time; the results are shown in Figure 1. The results show that the ratio of number of time slots over the number of tags obtains its minimum value around 2.32, when the ration of *TSC* over the number of tags is 0.88. So if we identify staying tags using ABS in f_i, we require to spend $2.32 \times |N_{i-1}| \times t$ time slots assuming the staying ratio is *t*. When using PRB, the time slots required depends on only the number of tags in the last frame $|N_{i-1}|$, which is $0.5 \times |N_{i-1}|$. Solving $2.32 \times |N_{i-1}| \times t < 0.5 \times |N_{i-1}|$, we obtain *changingRatio* as 0.2155.

There are some differences for ABS and PRB in estimating the number of tags. ABS requires exact estimation of the number of staying tags, while PRB can use threshold estimation to terminate the estimation early before the first empty slot appears, i.e., when $i > \log_2 \frac{preTagNum \times 0.2155}{1.2897} + 1$. Under these circumstances, the staying ratio is greater than *changingRatio*,



(a) n = 10 (b) n = 50 (c) n = 100

Fig. 1. Relation between the number of time slots and *TSC* in ABS

and PRB is used for tag identification. For ABS, the exact estimation is still required.

The framework of TAD is shown in the Algorithm 1. The variable *method* denotes that ABS or PRB will be used for further identification. Lines 4-7 show the early termination of estimation of tags and PRB will be used. Lines 8-12 show the exact estimation of tags and ABS will be used. For example, suppose that 50 tags were identified in f_{i-1}. Therefore, *preTagNum* is 50 and $\log_2 \frac{preTagNum \times 0.2155}{1.2897} + 1 = 4.06$. In f_i, reader estimates the number of staying tags. The reader monitors the value of *i* when tags reply. When *i* is increased to 5, the condition of line 4 of Algorithm 1 is satisfied. No further estimation is required and PRB will be used. When the fourth time slot is found to be an empty slot (i.e., $i = 4$), the estimation is normally terminated and ABS will be used.

The detailed identification algorithm is shown in Algorithm 2. When using ABS, the reader sets *TSC* to $\lceil 0.88 \times (arrEst + stayingEst) \rceil$ and broadcasts *TSC* to tags (lines 1-3 of reader operations). All tags choose a random number between 0 and *TSC* as its *ASC* (lines 1-3 of tag operations), and then follow ABS to communicate with the reader (line 4 of reader operations and line 4 of tag operations). In this approach, all the tags are identified equally. When using PRB, the reader sets *TSCEXT* to $\lceil 0.88 \times arrEst \rceil$ and broadcasts *TSC* and *TSCEXT* to the tags (lines 6-8 of reader operations). All arriving tags choose a random number between *TSC* to *TSCEXT* as their *ASC*s, and the staying tags keep their own *ASC*s (lines 6-10 of tag operations). Then the reader and tags follow PRB to communicate (lines 9-10 of reader operations and line 10 of tag operations). The staying tags and arriving tags are identified separately.

D. Discussions

Because of the different characteristics of time slots, different time durations are possible for different time slots [15]. For example, 6-bit transmission time is required for empty slots, and 99-bit transmission time is required for readable slots and collision slots. We can improve TAD considering such differences of time slots. ABS and PRB use the same improvements proposed by SSRB [15] when the algorithm is adapting to the staying ratio. The *changingRatio* can be similarly computed.

Algorithm 1: Estimate the number of staying tags

Reader Operations:

1 **for** $i = 1$ **to** tagID's length **do**
2 transmit i
3 receive tag responses
4 **if** *preTagNum* $\neq 0$ ***and*** $i > \log_2 \frac{preTagNum \times 0.2155}{1.2897} + 1$ **then**
5 *method* = '*PRB*'
6 **break**
7 **end**
8 **if** no tag responses **then**
9 *stayingEst* = $1.2897 \times 2^{i-1}$
10 *method* = '*ABS*'
11 **break**
12 **end**
13 **end**

Tag Operations :

1 receive reader's i
2 **if** i = the location of tagID's the rightmost bit 1 ***and*** *isArr* $= 0$ **then**
3 transmit ID
4 **end**

IV. Performance Analysis

In this section, we analyze the execution time of TAD. The identification of TAD can be represented as a traversal of a binary tree where each node represents a set of tags split by ASC. The inner nodes of the tree denote the collision slots, and the leaf nodes denote readable slots and idle slots.

In a binary tree of the depth k, there are 2^k nodes. If a collision happens, a tag's ASC randomly increases by zero or one. So it has $\frac{1}{2^k}$ probability to reach a leaf node after k collisions. In the tree, a leaf node is an idle slot when all n tags do not chooses that time slot. The probability is $\left(1 - \frac{1}{2^k}\right)^n$. A leaf node is a readable slot when one tag chooses this slot and the other tags do not. Its probability is $C_n^1 \left(1 - \frac{1}{2^k}\right)^{n-1} \frac{1}{2^k}$. The number of idle and readable slots in the depth k of the binary tree can be obtained by multiplying their probabilities by the number of nodes 2^k. Let $I(n)$ and $R(n)$ denote the number of idle slots and readable slots in a binary tree with n tags, respectively. They can be obtained as follows:

$$I(n) = \sum_{k=0}^{\infty} 2^k \left(1 - \frac{1}{2^k}\right)^n \tag{1}$$

Algorithm 2: Identify tags

Reader Operations:

if *method* = *'ABS'* **then**
  $TSC = \lceil 0.88 \times (arrEst + stayingEst) \rceil$
  transmit *TSC*
  use ABS to identify tags
end
if *method* = *'PRB'* **then**
  $TSCEXT = \lceil 0.88 \times arrEst \rceil$
  transmit *TSC* and *TSCEXT*
  $TSC = TSCEXT$
  use PRB to identify tags
end

Tag Operations :

if *method* = *'ABS'* **then**
  receive reader's *TSC*
  ASC = a random number between 0 and *TSC*
  follow ABS to communicate with the reader
end
if *method* = *'PRB'* **then**
  receive reader's *TSC* and *TSCEXT*
  if *isArr* = *1* **then**
    ASC = a random number between *TSC* and *TSCEXT*
  end
  follow PRB to communicate with the reader
end

$$R(n) = \sum_{k=0}^{\infty} n \left(1 - \frac{1}{2^k}\right)^n \tag{2}$$

Considering that the time slots consist of idle slots, readable slots, and collision slots. The number of collision slots in a tree $C(n)$ can be obtained by iterating its depth k from 0 to ∞:

$$C(n) = \sum_{k=0}^{\infty} 2^k \left\{1 - \left(1 - \frac{1}{2^k}\right)^n - n\frac{1}{2^k}\left(1 - \frac{1}{2^k}\right)^{n-1}\right\} \tag{3}$$

The number of nodes of the tree $A(n)$ (i.e., the number of time slots for identifying n tags) can be obtained as follows:

$$A(n) = C(n) \times 2 + 1 \tag{4}$$

In a dynamic environment, we assume that there are α arriving tags and β staying tags (i.e., there are $n - \alpha$ leaving tags). TAD first estimates the number of arriving tags and staying tags and then uses ABS or PRB for identification according to the estimation result. We denote estimated arriving tags and staying tags as $\hat{\alpha}$ and $\hat{\beta}$, respectively. The estimation is based on slot based LoF approach, and the number of time slots used in estimation is denoted by γ.

When the number of staying tags is sufficient large, TAD uses PRB for identification. We denote such approach as TAD_{PRB}. In the stage of identifying the staying tags, it consumes $\lceil \frac{n}{2} \rceil$ time slots. We denote optimal ratio of the number of initial slots over the number of tags as w. In the phase of identifying the arriving tags, TAD uses $\lceil w\hat{\alpha} \rceil$ time slots. For each slot, the probability that there are x tags replying in it is $C_{\alpha}^{x}\left(\frac{1}{\lceil w\hat{\alpha}\rceil}\right)^x\left(1 - \frac{1}{\lceil w\hat{\alpha}\rceil}\right)^{\alpha - x}$. We accumulate the probability when x changes from 0 to α, and obtain the number of time slots required for the identification:

$$A_{TAD-PRB}(\alpha + \beta) = \gamma + \lceil \frac{n}{2} \rceil + \lceil w\hat{\alpha} \rceil \sum_{x=0}^{\alpha} C_{\alpha}^{x}\left(\frac{1}{\lceil w\hat{\alpha}\rceil}\right)^x\left(1 - \frac{1}{\lceil w\hat{\alpha}\rceil}\right)^{\alpha - x} A(x) \tag{5}$$

When the number of staying tags is sufficient small, TAD use ABS for identification. We denote such approach as TAD_{ABS}. There are $\lceil w(\hat{\alpha} + \hat{\beta}) \rceil$ time slots initialized for $\alpha + \beta$ tags. We calculate the number of time slots required for the identification:

$$A_{TAD-ABS}(\alpha + \beta) = \gamma + \lceil w(\hat{\alpha} + \hat{\beta}) \rceil \sum_{x=0}^{\alpha+\beta} C_{\alpha+\beta}^{x} \left(\frac{1}{\lceil w(\hat{\alpha} + \hat{\beta})\rceil}\right)^x\left(1 - \frac{1}{\lceil w(\hat{\alpha} + \hat{\beta})\rceil}\right)^{\alpha+\beta-x} A(x) \tag{6}$$

If time slots are of different lengthes, we can compute the time duration for identifying n tags as follows. Suppose that the time duration of a collision slot, readable slot, and idle slot is *TC*, *TR*, and *TI*, respectively. The total time duration for identifying n tags is

$$T(n) = C(n) \times TC + R(n) \times TR + I(n) \times TI \tag{7}$$

V. Performance Evaluation

We evaluate the performance of TAD by comparing it with existing state-of-the-art algorithms including tree-based algorithms ABS, SRB, PRB, AQS [11], SSRB, FSA-CSS and aloha-based algorithm EDFSA (Enhanced Dynamic Framed Slotted ALOHA) [6]. A hundred of simulations are repeated to obtain each data point of the figures.

The experimental environment are as follows: In the previous frame f_{i-1}, there are n tags that are identified and all of them are arriving tags (i.e., no staying tags). After that, tags may come in or out the interrogation region of the reader. Our purpose is to minimize the time to identify all the tags present in the interrogation region of the reader in frame f_i. l denotes the length of tag ID. Each bit is transmitted for $b\mu$s. We set l=96 bits, n=50, and b=5 to calculate the execution time.

A. Theoretical and Actual Number of Time Slots in TAD

In section IV, we analyzed the time slots required for TAD. We first compare the computed theoretical values with real values in the simulations. We set arriving ratio to 0.2 and change staying ratio to check the performance. The result is shown in Fig. 2. As shown in the figure, when the staying ratio increases from 0 to 1, the theoretical value and actual value are quite close. The average deviation from the theoretical value to actual value is 8.57% (7.5773 time slots), which is a small value probably due to the estimation errors of tag number. This confirms the correctness of the computation in the performance analysis of TAD.

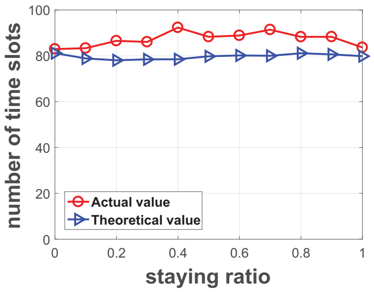


Fig. 2. Theoretical and actual number of time slots in TAD

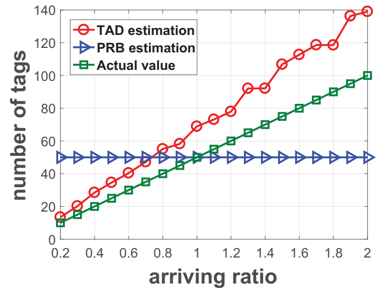


Fig. 3. Tag estimation accuracy of TAD and PRB

B. Tag Estimation Accuracy

We compare the tag estimation performance of TAD and PRB. The staying ratio is set to be 1, therefore the number of tags present depends only on the arriving ratio. We change the arriving ratio to check the performance. The result is shown in Fig. 3. As shown in the figure, the actual number of tags increases linearly. PRB always estimates the tags as the number of tags in the previous frame, 50 tags. TAD uses the feedback of current existing tags to estimate the number of tags and hence more accurate. The average deviation between the TAD estimation and the actual value is 36.71% while that between the PRB estimation and the actual value is 71.59%.

C. Impact of Arriving Ratio

We fix staying ratio and vary arriving ratio to compare the execution times of different algorithms; the results are shown in Fig. 4(a)-4(c). PRB is always better than SRB since PRB spends half of time slots of SRB to identify the staying tags. When the staying ratio is 0, ABS always outperforms PRB (see Fig. 4(a)); and when the staying ratio is 0.5, ABS outperforms PRB if the arriving ratio is less than 0.3 (see Fig. 4(b)). While the staying ratio is 1, ABS outperforms PRB when the arriving ratio is less than 0.2 (see Fig. 4(c)). PRB causes many idle slots if there are few arriving tags. This is because PRB estimates the arriving tags the same with that in the previous frame and then reserve time slots for them. On the contrary, ABS does not reserve time slots for arriving tags but identify all the tags equally. PRB outperforms AQS in all cases because AQS requires to send a prefix in each query, and the pair resolution is not used. Among the above approaches, PRB and ABS achieve the best performance in different conditions, showing necessary to sense the arriving/staying ratios and perform PRB or ABS adaptively. TAD adopts such idea to design its hybrid strategy.

It can be seen that in most of time TAD and SSRB outperform other algorithms. This is because when identifying staying tags, a 3-bit symbolic response is transmitted in TAD and SSRB. Besides, when identifying arriving tags, the idle slots of TAD and SSRB are much shorter than other time slots. When staying ratio is 0, TAD outperforms SSRB if the arriving ratio is less than 0.6 (see Fig. 4(a)); and when arriving ratio is 0.5 or 1, TAD outperforms SSRB if the arriving ratio is less than 0.5 (see Fig. 4(b) and Fig. 4(c)). This is because TAD is able to identify two staying tags in one slot. However, with the increase of arriving tags, the estimation errors of TAD increases. This is consistent with the results in previous subsections. FSA-CSS and EDFSA do not distinguish staying tags from arriving tags. The identification is from the scratch even in the repeated identifications. They perform not well when the staying ratio is large.

D. Impact of Staying Ratio

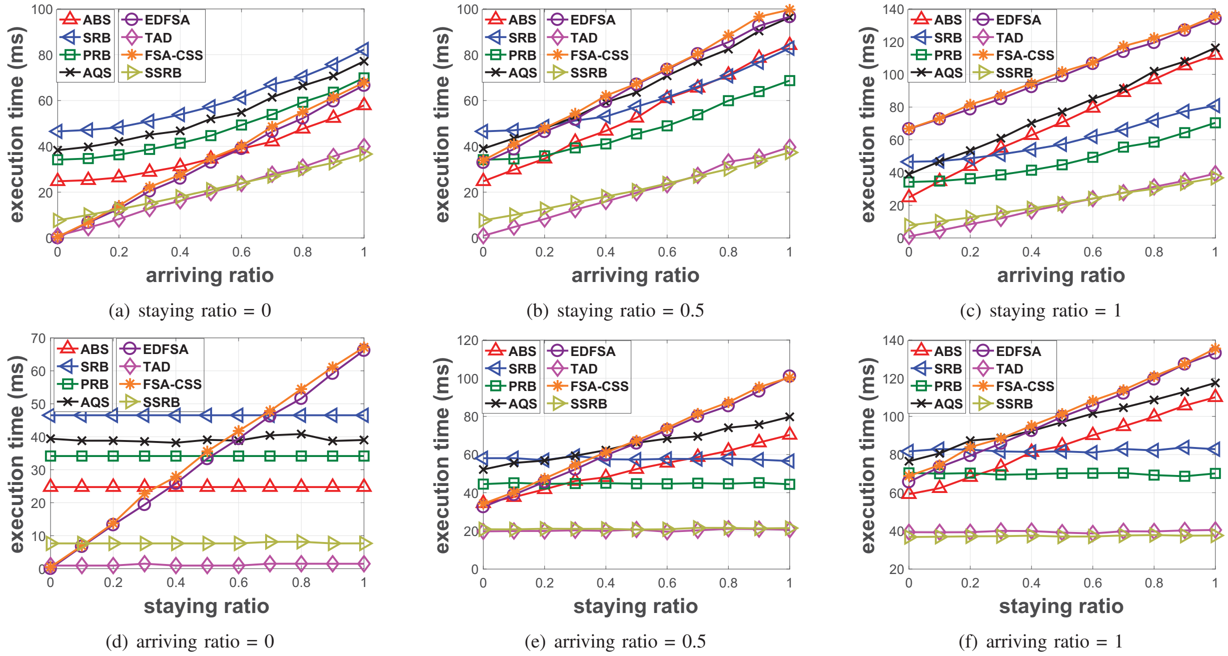
Then, we vary the staying ratio to check the execution times of different algorithms; the results are shown in Fig. 4(d)-4(f).

In these figures, the execution times of SRB and PRB do not change. This is because SRB and PRB always spend fixed number of time slots in identifying staying tags, and then staying ratio does not affect their execution time. PRB is always better than SRB since PRB spends half of time slots of SRB to identify the staying tags. When the arriving ratio is 0, ABS always outperforms PRB (see Fig. 4(d)); and when the arriving ratio is 0.5 or 1, ABS outperforms PRB if the staying ratio is no more than 0.3 (see Fig. 4(e) and Fig. 4(f)). This is because PRB always reserves time slots for arriving tags even if arriving ratio is small while ABS does not. AQS always underperforms ABS and PRB since AQS broadcasts a prefix in each time slot and hence the amount of data sent is more than that of ABS and PRB.

In most of cases, TAD and SSRB always outperform other algorithms (see Fig. 4(e) and Fig. 4(f)). This is because TAD and SSRB use shorter slots when identifying tags. The execution times of TAD and SSRB are stable (see Fig. 4(d), Fig. 4(e), Fig. 4(f)). This is because the two algorithms all consume constant number of time slots in identifying staying tags. TAD outperforms SSRB when arriving ratio is 0 or 0.5, but underperforms SSRB when arriving ratio is 1. It shows that TAD estimates the cardinality of tags more accurately if the number of tags is small. TAD estimates the arriving tags as $1.2897 \times 2^{i-1} (i = 1, 2, \ldots)$, which is more accurate when the arriving ratio is small.

VI. Conclusion

In the paper, we propose a new tag anti-collision algorithm called TAD for dynamic environment. TAD automatically adapts to different staying ratios, by using different identification approaches. A threshold estimation is proposed



(a) staying ratio = 0
(b) staying ratio = 0.5
(c) staying ratio = 1
(d) arriving ratio = 0
(e) arriving ratio = 0.5
(f) arriving ratio = 1

Fig. 4. Execution times of different approaches by varying arriving ratio or staying ratio

especially for speeding up the staying ratio estimation for the hybrid identification approach. The simulations show that TAD outperforms the existing approaches in the dynamic environment.

Acknowledgments

This research is supported in part by National Natural Science Foundation of China No. 61502351, National Key R&D Program of China No. 2018YFC1604000, and Chutian Scholars Program of Hubei, China.

References

[1] M. Shahzad and A. X. Liu, "Probabilistic optimal tree hopping for rfid identification," *IEEE/ACM Transactions on Networking*, vol. 23, no. 3, pp. 796–809, 2015.

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

[3] J.-l. Zheng, T.-f. Qin, and G.-n. Ni, "Tree-based backoff protocol for fast RFID tag identification," *The Journal of China Universities of Posts and Telecommunications*, vol. 20, no. 2, pp. 37–41, 2013.

[4] Y.-C. Lai, L.-Y. Hsiao, and B.-S. Lin, "Optimal slot assignment for binary tracking tree protocol in RFID tag identification," *IEEE/ACM Transactions on Networking (TON)*, vol. 23, no. 1, pp. 255–268, 2015.

[5] G. Maselli, C. Petrioli, and C. Vicari, "Dynamic tag estimation for optimizing tree slotted aloha in RFID networks," in *Proc. of International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2008, pp. 315–322.

[6] S.-R. Lee, S.-D. Joo, and C.-W. Lee, "An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification," in *Proc. of International Conference on Mobile and Ubiquitous Systems: NETWORKING and Services*, 2005, pp. 166–174.

[7] J. Park, Y. C. Min, and T. J. Lee, "Identification of RFID tags in framed-slotted ALOHA with tag estimation and binary splitting," in *Proc. of International Conference on Communications & Electronics*, 2007, pp. 368–372.

[8] P. Solic, J. Radic, and N. Rozic, "Early frame break policy for ALOHA-Based RFID systems," *IEEE Transactions on Automation Science & Engineering*, vol. 13, no. 2, pp. 876–881, 2016.

[9] X. Liu, K. Li, J. Wu, A. X. Liu, X. Xie, C. Zhu, and W. Xue, "Top-k queries for multi-category RFID systems," in *Proc. of IEEE International Conference on Computer Communications*, 2016, pp. 1–9.

[10] W. T. Chen, "Optimal frame length analysis and an efficient anticollision algorithm with early adjustment of frame length for RFID systems," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3342–3348, 2016.

[11] J. Myung, W. Lee, and J. Srivastava, "Adaptive binary splitting for efficient RFID tag anti-collision," *IEEE communications letters*, vol. 10, no. 3, pp. 144–146, 2006.

[12] Y.-C. Lai and C.-C. Lin, "Two blocking algorithms on adaptive binary splitting: single and pair resolutions for RFID tag identification," *IEEE/ACM Transactions on Networking (TON)*, vol. 17, no. 3, pp. 962–975, 2009.

[13] J. Su, Z. Sheng, L. Xie, G. Li, and A. X. Liu, "Fast splitting based tag identification algorithm for anti-collision in UHF RFID system," *IEEE Transactions on Communications*, 2018.

[14] J. Su, D. Hong, J. Tang, and H. Chen, "An efficient anti-collision algorithm based on improved collision detection scheme," *IEICE Transactions on Communications*, vol. 99, no. 2, pp. 465–470, 2016.

[15] R. Jayadi, Y.-C. Lai, and C.-C. Lin, "Efficient time-oriented anti-collision protocol for RFID tag identification," *Computer Communications*, vol. 112, pp. 141–153, 2017.

[16] Y. Jiang, R. Zhang, W. Cheng, B. Li, and W. Sun, "An adaptive ternary query splitting based tag anti-collision protocol for mobile rfid systems," in *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*. IEEE, 2016, pp. 1–6.

[17] Y.-C. Lai and C.-C. Lin, "Two couple-resolution blocking protocols on adaptive query splitting for RFID tag identification," *IEEE transactions on mobile computing*, vol. 11, no. 10, pp. 1450–1463, 2012.

[18] C. Qian, H. Ngan, Y. Liu, and L. M. Ni, "Cardinality estimation for large-scale RFID systems," *IEEE transactions on parallel and distributed systems*, vol. 22, no. 9, pp. 1441–1454, 2011.

[19] J.-B. Eom, T.-J. Lee, R. Rietman, and A. Yener, "An efficient framed-slotted ALOHA algorithm with pilot frame and binary selection for anti-collision of RFID tags," *IEEE Communications Letters*, vol. 12, no. 11, 2008.