Hash Based Collision Resolution for Passive RFID Tags

Wei Yen
Department of Computer Science and Engineering, Tatung University
wyen@ttu.edu.tw

Abstract*-RFID technology has recently attracted renewed interest due to the major promotion from global retailers. In this paper, we consider the tag collision problem that prevents the reader from identifying the adjacent tags in certain applications. We propose a hash based algorithm that requires neither memory nor battery in the tag. Only simple arithmetic capability is necessary. The proposed algorithm guarantees complete tag reading while reserving the power in the reader. Using the information collected by the reader, it increases the speed of repeated readings of the tags moving in and out of the reading zone. We compare our algorithm with two algorithms based on query-tree searching and on dynamic reading frames. We show that the proposed algorithm is the most effective in terms of the detection speed and number of reader messages.*
Keywords: RFID, collision resolution, query-tree, hash algorithms

1. Introduction

Although Radio frequency identification (RFID) technology has become a practical tool since early 1980's [1], it has recently received a major push from renowned corporations worldwide. It is envisioned that once the economic scale is reached, RFID will create new ways of operations and reduce the cost for companies and consumers. A RFID system can be used for automatic identification, object tracking, inventory counting, customized marketing, and many other applications still being created. The main components of a RFID system are a reader, RFID tags, and related management and/or database subsystems (Figure 1). The reader emits radio signals to obtain the identifications of the tags in its proximity. The electromagnetic energy received by the tags, hence, powers the associated chips. With the received reading signal, the chip decides if the tag should respond to the reader. If the answer is positive, then the tag sends the related information, usually its identification, to the reader. In one implementation, the reading signal specifies a range of IDs. A tag will respond to the reading signal if its ID falls into the range. Depending on various applications, RFID systems come in different shapes and forms. For example, the reader can be embedded in the doorway with the reading range of a quarter to half meter in a security application. Or, the tags can be equipped with memory and battery to maintain transaction data.

Currently, there are several organizations working on the standardization of RFID technology. RFID experts group (REG) has formed early this year within Association of Automatic Identification and Mobility (AIM) Global to grapple with issues of usage parameter definition to packaging to recyclability. EPCglobal backed by Uniform Code Council (UCC) has published several specifications. They address the transmission spectrum, reader-tag messaging protocol, and the support of the existing identification
systems.

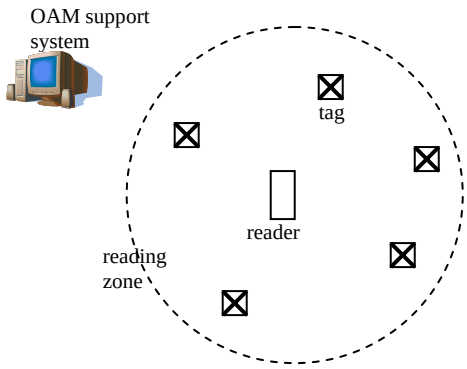

Figure 1. An example of RFID system

There are potentially two media access (MAC) layer issues in the RFID system. They are referred to as the reader collision and tag collision problems. If two readers are within each other's reading range and try to read tags at the same time, then the tags may react strangely or not respond at all. This is called the reader collision problem. However, if several tags respond simultaneously to a reading signal, the tag collision problem occurs. In this case, the reader will not be able to tell the identities of these responding tags. In this paper, we focus on the tag collision problem.

The remainder of this paper is structured in the following manner. In Section 2, we will highlight the existing works in this area. Then, we will formally define the problem and propose our solution in Section 3. The performance of the proposed algorithm is discussed in Section 4. We

conclude the paper and list possible directions for future study in Section 5.

2. Existing and Related Works

In this section, we discuss the existing works in solving the tag collision problem. In addition, we introduce a related work on the reader collision problem.

In [2], a query-tree based algorithm is proposed to govern the order of transmissions among the tags. In this algorithm, the RFID reader follows a binary query tree to inquire the tags. If the prefix of a tag's ID matches what is sent by the reader, then the tag responds. If there is only one such tag, then the reader recognizes the tag and no collision needs to be resolved. However, when there are two or more tags with the same prefix, all respond and the reader learns a collision from the garbled message. The reader can resolve the collision by being more and more precise on its query, i.e., specifying more bits in the prefix.

In this algorithm, the tag is stateless. In other words, it does not have to remember if it already responds to the reader in a reading cycle. The only operation required is the comparison of its ID. Eventually, all tags will be successfully read exactly once since their IDs are unique. The algorithm has its root at the well studied packet radio network and its theoretic time complexity is well understood. However, the query-tree algorithm is not very efficient as far as the reader's power is concerned. Since the reader is usually operates on battery, we wish to reduce the number of messages it sends to increase the battery life and reduce the system cost.

In [4], the algorithm using dynamic reading frames is proposed. In this algorithm, the reader initiates a reading cycle by sending a query containing an ID range, a random number, and the size of the reading frame (N). After the query is sent, a reading frame of N slots follows. A tag whose ID is in the range uses the random number to determine on the slot in which the tag transmits its data. The frame size N is updated by the reader according to the number of occupied and collision slots in the previous frame (Equation (1)). In general, N stabilizes as it reaches the number of tags in the system.

$$N_{next} = N_{occupied} + 2N_{collision} \quad (1)$$

where N_{next} is the size of the next reading frame, $N_{occupied}$, $N_{collision}$ are the numbers of occupied and collision slots, respectively.

This algorithm is believed to be faster than the query-tree algorithm since it tries to read all tags at once rather than to resolve individual collisions. However, this algorithm cannot guarantee that all tags in the reading zone are identified. This shortcoming results from the loose control of the reader on when the tags transmit. This problem may be critical to certain applications. Another potential issue is the size of N may be limited because passive tags rely on the power provided by the reader for their transmissions. It is not clear how the algorithm adapts to the cases where the number of tags, n, is much greater than the maximum frame size N_{max}.

There is a significant drawback in both algorithms. The system cannot take advantage of the collected information about the tags. For example, the tags in a group are read already by the reader. After a short interval, tags move in and out of the reading zone. Suppose the number of new tags joining and of old tags leaving the system is relatively small when compared with the total number of tags in the zone. In both algorithms, it will cost the system the same to identify the tags in the group regardless of their being read before. Our proposed algorithm is designed to address this issue as well.

Waldrop et al. proposed Colorwave to address the reader collision problem [5]. Colorwave is enhanced from the distributed color selection algorithm. The readers take turns to query the tags in an interval. If two or more readers interfere with one another, then the kick process starts and each reader involved will find a new slot in the interval. Colorwave allows the interval to adjust dynamically based on the utilization. For instance, if a reader is usually successful on sending out its query, it will start to reduce the size of the interval. While Colorwave dynamically adapts to the population of tags, the 'color' assignment tends to fluctuate in certain conditions. The effect of this fluctuation is not clear. Although the reader collision problem is not in the focus of this paper, we need to be aware of the issues in multi-reader-multi-tag environment so that our solution is extendable.

Next, we propose the hash based collision resolution method that avoids the aforementioned problems of the query-tree and dynamic frame algorithms. The algorithms in both tags and reader will be explained.

3. Hash Based Collision Resolution

The tag collision problem is very similar to the MAC problem in the traditional packet ratio network. There is one shared communication channel and if two tags send at the same time, then the messages become useless. However, the RFID system exhibits several distinctive characteristics that are not observed in the packet radio network. As an example, the traffic of a radio source can be locally generated or relayed data. Yet, the tag's transmission is always triggered by the reader. Generally speaking, the tag possesses limited processing and storage capability. In addition, the

purpose of the RFID system is for the reader to find out the IDs of all tags in its reading zone while that for the packet radio network is to transfer data. In the following, we list the design objectives of our algorithm.

- No memory in the tag is required
- Limited processing capability at tags
- Fast detection of addition and deletion of tags
- Reduce the number of reading messages
- Fast reading
- Guaranteed identification

Next, we will describe the system parameters and the proposed hash based algorithm in detail.

Suppose there are n RFID tags and a single reader. Each tag has its own unique binary encoded ID, d_i. The reader sends reading messages to poll the IDs. Every reading message contains several integer pairs. An integer pair is comprised of a base number, p_i, and an integer, l_i. Let P_i denote the set of the integer pairs transmitted in the i^{th} polling. The tags compute if and when they should send their IDs in the reading frame based on P_i. A complete reading cycle includes the reading message and the associated reading frame. The time intervals for these two portions are denoted by, t_m and t_f, respectively. The algorithm for this tag computation is listed in the following.

```
Tag_Slot(P_i, d_j)
    m ← | P_i |
    c ← d_j
    for k = 1 to m
       q_j ← (c – l_k) / p_k
       r_i ← (c – l_k) mod p_k
       if k ≠ m and r_i ≠ 0
          then return
       c ← q_i
    s_i ← r_i+1
    transmit d_j at the s_i-th slot
    in the reading frame
```

Figure 2. Computing tag transmission slot

This algorithm first makes sure if $(d_j\text{-}l_1)$ is completely divisible by p_1. If so, the quotient is used as the dividend of the next division. p_2 and l_2 are the divisor and expected remainder of the next division. This process continues till the tag figures out if and when to transmit its ID. If a remainder in one of the divisions (except for the last one) of this process is different from what is specified (l_i), the tag will not transmit in the reading cycle. However, if the remainders are as expected, then the remainder of the last division indicates the slot when the tag will send its ID. Using **Tag_Slot**(), the tag with d_i = 31259 will transmit at 8^{th} slot if P_i = {(61, 27), (61, 24), (61, 0)} in this algorithm. As another example, if d_i = 43884 and P_i = {(17, 7), (17, 15), (17, 0)}, then the tag will not transmit its ID in this reading cycle at all. As Figure 2 shows, the computational complexity for the tag is low (O(log d_i)) and can be implemented at reasonable cost.

Given the tag's capability, the binary query-tree algorithm can be extended to Q-ary tree if we fix all p_i to a selected number in the reading messages. The performance of Q-ary query tree is out of the scope of this paper. We merely point out that our algorithm can easily be modified to support query-tree based algorithms. In the following, we focus the discussion on the reader algorithm.

In the proposed algorithm, we assume the maximum reading frame contains N_{max} slots. The reader adopts different methods when reading the tags. When a group of tags are first identified by the reader, the probing method is invoked. However, if the reader has processed the tags previously or knows the number of tags in the neighbor collision slot, then the confirmation method is used. The main difference between these two methods is that the former adjusts the base number according to the number of collision slots. The latter, however, uses the acquired information to pick base numbers.

In the proposed algorithm, the reader resolves the collisions following a specific order. In essence, the reader will send the initial reading message and observe the associated reading frame. As shown in Figure 3, the reader records all slots with contention and tries to resolve the collisions in depth first manner. This is the technique applied in both [2] and [3]. The reader can target the tags jamming a particular slot by indicating the l_i in the integer pair. Next, we explain how the integer pairs in the reading messages are determined.

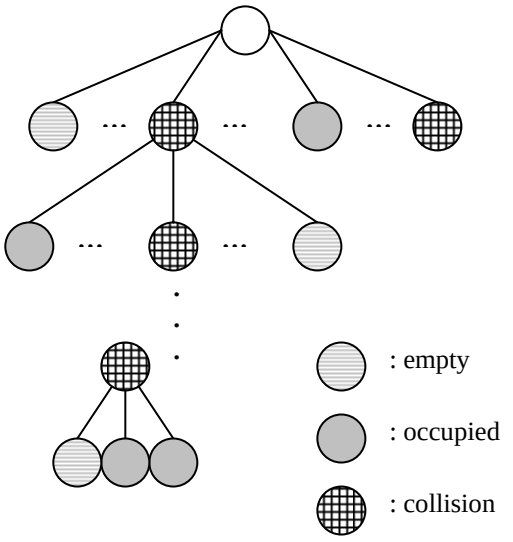

Figure 3. Order of Reading Messages

In the probing method, the reader has no idea how many tags are in its reading zone. Hence, it sends P_1 = {(p_1, 0)} to the tags where p_1 is the largest prime number smaller than N_{max}. For instance,

if N_{max} is 64, then p_1 is 61. If the number of tags, n, is smaller or close to p_1, then there should not be too many slots with collisions. Therefore, based on the number of collided slots, the algorithm determines the base number it uses to resolve the collision in the next layer of the query tree.

Reading message	slots w/ collisions	occupied slots
P_1: {(13,0)}	3, 4, 10, 11	1, 8, 12
P_2: {(13,2), (7, 0)}	2, 7	5
P_3: {(13,2), (7, 1), (3, 0)}	-	2, 3
P_4: {(13,2), (7, 6), (3, 0)}	-	1, 2
P_5: {(13,3), (5, 0)}	4	2, 4, 5
P_6: {(13,3), (5, 3), (2, 0)}	-	1, 2

Figure 4. Example of the probing method

Specifically, we can define the level of contention using the percentage of the collided slots in the reading frame. If less than 50% of the slots contain collisions, we pick a new base number that is both a prime number and about half of the current one. The current base number remains if the contention level is equal to or greater than 50%. We wish to remark that a good estimation of the number of tags involved in a collision will improve the performance of the algorithm. To further enhance the algorithm, we can follow the procedure just described and find out the number, say x, of tags in a collision. Then, when the reader attempts to resolve the collision in the sibling slots in the query tree, this closest prime to this number x can be used as the base number.

An example of the probing method is illustrated in Figure 4. In this example, N_{max} is 16 and p_1 is 13. The reader observes that slots 3, 4, 10, and 11 contain collisions. And, the reader will resolve these collisions in order. Since the contention ratio is less than 50%, the base number is reduced to 7 in P_2. The reading messages P_2 to P_4 are sent to solve the collision in slot 3 of the first reading message. And, the reader discovers that there were five tags colliding in the slot. This information is used to set the base number for resolving the collision in slot 4 of the first reading message.

Since tags can move in and out of the reading zone of the reader, it is an interesting improvement if we can reduce the reading time and/or message complexity for the previously read tag group with some changes. In theory, once all tags are identified, the reader can figure out the optimal sequence of reading messages for these tags. A greedy algorithm may be designed to achieve this goal. This mode of reader operation is referred to as the confirmation method in the proposed algorithm. This confirmation method characterizes this proposed algorithm since the other two algorithms do not exhibit this capability.

If the number of tags, n, is much smaller than N_{max}, the confirmation method finds a based number that is smaller than n and maximizes successful tag transmissions. In fact, the method aims to optimize successful tag transmissions in each known collision resolution. For unexpected collisions, the based number is selected using the average tag transmissions in the sibling slots.

The confirmation method changes its tactics if the number of tags is closer to or even greater than N_{max}. In such cases, the reader will initially behave as the probing method is applied. Then, to solve the contention in the initial reading frame, the reader will intend to increase the successful transmissions of tags based on the information already collected. Again for unexpected collision, the average number of tags in the sibling slots is used as reference to pick the base number.

We have, so far, described the hash based distributed algorithm for the RFID system. In the following section, we investigate the performance of the proposed algorithm and compare it with that of the query-tree and dynamic reading frame algorithms.

4. Performance Evaluation

In this section, we compare the proposed algorithm with the query-tree and dynamic frame algorithms. Throughout the simulations, we set initial frame size of the dynamic frame algorithm to 32.

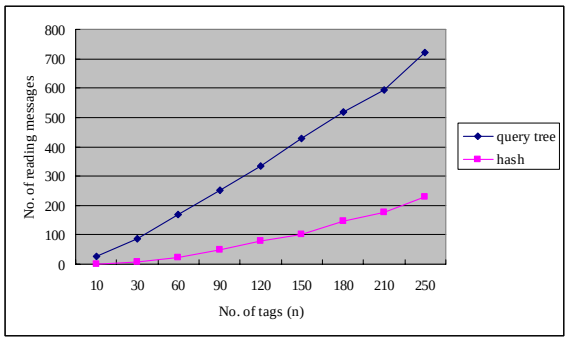

Figure 5. Number of reading messages required

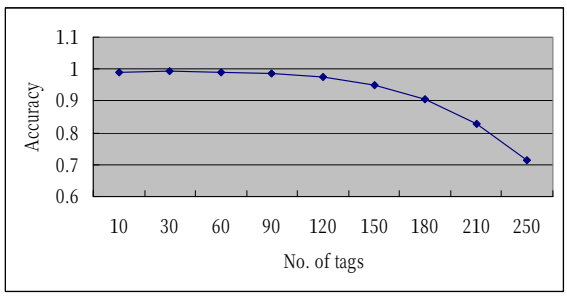

Figure 6. Rate of accurate identification for dynamic frame algorithm

From Figure 5 to Figure 8, we set the maximum frame size to 128 and let each tag has 9-digit ID. And, the ID numbers are distributed uniformly.

Then, we change the number of tags in the reading zone from 10 to 250 and observe the results. In Figure 5, we see the number of reader messages required in the query tree and the hash algorithms increases linearly as the number of tags increases. However, the query-tree algorithm generates three times as many reading messages as the hash based algorithm does. Figure 5 does not show the performance of the dynamic frame algorithm. This is because it always produces 10 reading messages in our simulation regardless the number of tags. As it grows, this algorithm starts to miss tags as shown in Figure 6. More than 10% of tags are not recognized by the algorithm, when the number of tags is greater than 170.

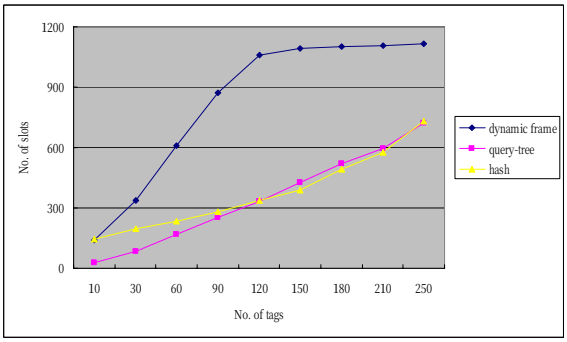

Figure 7. Number of reading slots generated

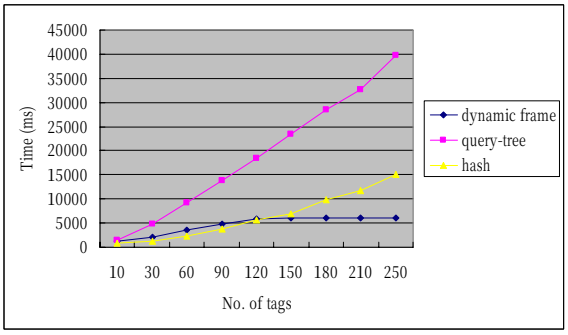

Figure 8. Total time for tags identification

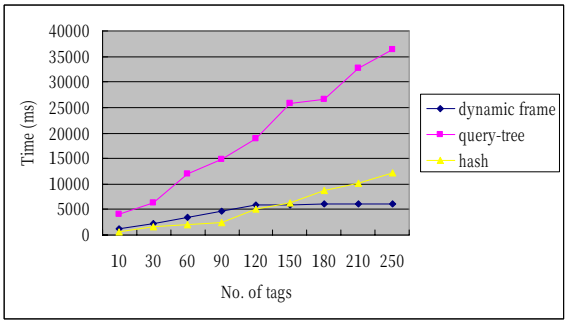

Figure 9. Total time for tags identification (with clustered tag IDs)

In Figure 7, we consider the number of slots in the reading cycle. Combining this figure and Figure 5 , we can calculate the reading time for all tags in the reading zone. For example, we can assume the reading message takes 50 ms to complete while a slot in the reading cycle lasts 5 ms. The result is illustrated in Figure 8 .

We note that although the query-tree and hash algorithms create about the same number of tag slots, the former needs much more reading messages. Hence, the hash algorithm is much faster than the query-tree algorithm. The curve for the dynamic frame algorithm becomes flatten as the number of tags exceeds N_{max}. This is because the size of the reading cycles remains close to N_{max} due to high percentage of occupied and collision slots. Please refer to Equation (1) for the calculation of the frame size.

Next, we repeat the same experiments but use clustered tag IDs. That is the tags now have IDs that are close to each other. This is to demonstrate the effect of another ID distribution on the performance. As Figure 9 shows, all three algorithms are relatively insensitive to this distribution adjustment. We would like to elaborate on this result because it seems reasonable to speculate the query-tree algorithm will suffer performance degradation. For example, the algorithm needs to send 9 reading messages to distinguish 01010100 and 01010101 but only two for 01010101 and 11010101. The simulation result suggests that for the same amount of tags, the algorithm spends most time on resolve collisions when the IDs are clustered. However, probing becomes the main task when the IDs are spread out. At the end, the algorithm performs roughly the same in these two situations.

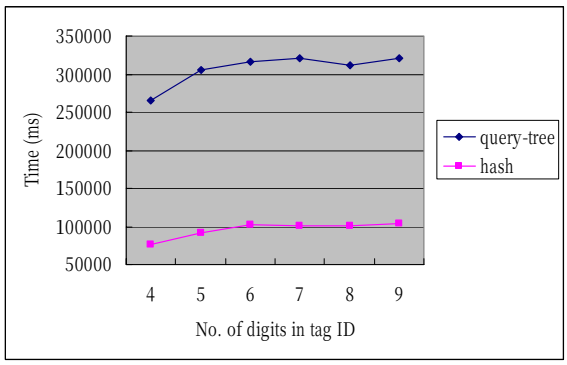

Figure 10. Total time for tags identification (with various numbers of digits of tag IDs)

In Figure 10, we uniformly select 2000 tags and vary the number of digits in the tag ID. As the result shows, the performance of the query-tree and hash algorithms stabilizes when there are 6 digits or more in the tag ID. The algorithms read faster when there are less digits. This is because the maximum depth of the searching tree is smaller. However, this effect diminishes if the number of tags is too small in the entire ID space.

5. Conclusions and Future Works

In this paper, we propose a hash based algorithm and show it has better performance than the query-tree and dynamic frame algorithms in speed, identification completeness, and reader power reservation. We also observe that clustered IDs do not change the performance of these algorithms drastically. To enable attendance-free check out, precise item identification is imperative. Although the hash algorithm is the best among the three, it along cannot handle items with the same IDs. We think the dynamic frame algorithm can be combined with the hash algorithm to solve this problem. There are other possible directions for future researches. First, we wish to explore the possibility of applying hash technique in breadth first manner. On the other hand, we will consider the reader collision problem. If the length of the reading cycle is not constant as in the dynamic frame and hash based algorithms, then the reader collision problem will become more complicated. Eventually, our goal is to develop distributed algorithms to make a multi-reader-multi-tag environment possible.

Acknowledgments

This work is supported by the Tatung University under grant number B93-I07-038.

References

[1] R. E. Floyd, "Radio Frequency Idenfication," *In the Proc. of IEEE Southeastcon '93*, Charlotte, NC, U.S.A., Apr. 1993.

[2] C. Law, K. Lee, and K.-Y. Siu, "Efficient Memoryless Protocol for Tag Identification," in the *Proc. of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, Boston, MA, United States, pp. 75-84, Aug. 2000.

[3] J. I. Capetanakis, "Tree Algorithms for Packet Broadcast Channels," *IEEE Transactions on Information Theory*, Vol. IT-25, No. 5, Sept. 1979.

[4] H. Vogt, "Multiple Object Identification with Passive RFID Tags," in the *Proc. of the IEEE International Conference on Systems, Man and Cybernetics*, Yasmine Hammamet, Tunisia, Vol. 3, pp. 651-656, Oct. 2002.

[5] J. Waldrop, D. W. Engels, and S. E. Sarma, "Colorwave: An Anticollision Algorithm for the Reader Collision Problem," *IEEE ICC 2003*, Anchorage, AK, United States, Vol. 2, pp. 1206-1210, May 2003.