Performance Evaluation of Anti-Collision Algorithms for RFID System with Different Delay Requirements

Warakorn Srichavengsup
Computer Engineering, Robotics and Technology Laboratory
Faculty of Engineering, Thai-Nichi Institute of Technology
1771/1 Pattanakarn Rd., Suanluang, Bangkok 10250, Thailand

Abstract—The main purpose of Radio-frequency identification (RFID) implementation is to keep track of the tagged items. The basic components of an RFID system include tags and readers. Tags communicate with the reader through a shared wireless channel. Tag collision problem occurs when more than one tag attempts to communicate with the reader simultaneously. Therefore, the second-generation UHF Electronic Product Code (EPC Gen 2) standard uses Q algorithm to deal with the collision problem. In this paper, we introduce three new anti-collision algorithms to handle multiple priority classes of tags, namely, DC, DQ and DCQ algorithms. The goal is to achieve high system performance and enable each priority class to meet its delay requirement. The simulation results reveal that DCQ algorithm is more effective than the DC and DQ algorithms as it is designed to flexibly control and adjust system parameters to obtain the desired delay differentiation level. Finally, it can conclude that the proposed DCQ algorithm can control the delay differentiation level and yet maintain high system performance.

Keywords—RFID; Anti-collision; Q algorithm; Priority

I. INTRODUCTION

Radio-frequency identification (RFID) uses radio-frequency electromagnetic fields to identify and track the objects [1]. The RFID system consists of tags and readers. The tags connect with the readers through a communication channel. During the identification process, each tag sends its identification (ID) code with a probability specified by a system to RFID reader. The EPC Gen 2 air-interface protocol [2-3] employs an anti-collision protocol called Q algorithm [4-7]. In the Q algorithm, each tag randomly selects an integer from the specified range [0, 2^Q-1]. Fig.1 demonstrates the flow chart of the Q algorithm. This algorithm defines a floating-point representation of Q, Q_fP, and a fixed step size, C. The RFID reader adjusts the Q_fP parameter based on the current slot state and then Q_fP is rounded to the nearest integer value, Q. When the collision [8] happens in the current slot, Q_fP increases by C. On the contrary, Q_fP decreases by C when the current slot is idle. The value of Q_fP remains unchanged when only one tag accesses the current slot. Nevertheless, the standard Q algorithm is not designed to support the RFID system with different delay requirements. Example of RFID system with different delay requirements is Automatic Vehicle Identification (AVI) system [9-14]. In the Automatic Vehicle Identification system, the tags that are attached to the emergency vehicles such as ambulances and fire trucks [15-19] needed to be identified before the other vehicles.

Therefore, the new anti-collision algorithms, DC, DQ and DCQ, which are suitable for the RFID system with different classes of tags, are developed. The purpose of these algorithms is to prioritize delay sensitive tags over the less sensitive tags while maintaining high system performance.

The paper is structured as follows. In section II, we shall describe the details of DC, DQ and DCQ algorithms. The results and discussion are presented in Section III. Finally, the conclusion is given in Section IV.

II. PROPOSED ANTI-COLLISION ALGORITHMS

In this paper, we modified the Q algorithm to the anti-collision algorithms that are capable of handling multiple priority classes, namely, DC, DQ and DCQ algorithms.

Let we first define the following parameters which are used in the detailed description of the algorithms:

\[ C_1 = \text{the step size for priority class 1 tags.} \]
\[ C_2 = \text{the step size for priority class 2 tags.} \]
\[ Q_1 = \text{the initial value of } Q_{fP} \text{ for priority class 1 tags.} \]
\[ Q_2 = \text{the initial value of } Q_{fP} \text{ for priority class 2 tags.} \]

A. DC algorithm

This algorithm is further developed from the Q algorithm to handle multiple priority classes. In the Q algorithm, a
predefined step size $C$ is set equally for all tags. Therefore, if the system supports tags with different priority classes, it is more suitable to set different predefined step sizes to different priority classes. This anti-collision algorithm is referred to as Different values of $C$ (DC) algorithm. The order of $Q$ parameter update operations of this algorithm is the same as the $Q$ algorithm, except that the step size values for priority classes 1 and 2 are equal to $C_1$ and $C_2$, respectively.

Fig. 2 illustrates the flow chart of DC algorithm. In this flow chart, when the collision happens, $Q_{fp}$ of the priority class 1 tags increases by $C_1$ whereas $Q_{fp}$ of the priority class 2 tags increases by $C_2$. When the current slot is idle, the values of $Q_{fp}$ for the priority class 1 and 2 tags decrease by $C_1$ and $C_2$, respectively.

Fig. 3 demonstrates the approaches for the DQ algorithm to adjust the $Q_{fp}$ values for different priority classes. The DQ algorithm follows the same steps as the $Q$ algorithm, except that the initial values of $Q_{fp}$ for priority classes 1 and 2 are equal to $Q_1$ and $Q_2$, respectively. This means that the initial frame lengths for priority classes 1 and 2 are equal to $2^{Q_1}$ and $2^{Q_2}$, respectively.

C. DCQ algorithm

This algorithm is obtained by combining the DC and DQ algorithms. In this algorithm, the system performance is achieved by assigning different values of $C_1$, $C_2$, $Q_1$ and $Q_2$ according to its delay requirement. This more flexible algorithm will be referred to as Different values of $C$ and initial $Q_{fp}$ (DCQ) algorithm.

The procedure of adjusting the values of $Q_{fp}$ can be seen in Fig. 4. The $Q$ parameter update operations of DCQ algorithm are the same as the $Q$ algorithm, except that the step sizes for priority classes 1 and 2 are equal to $C_1$ and $C_2$, respectively and the initial values of $Q_{fp}$ for priority classes 1 and 2 are equal to $Q_1$ and $Q_2$, respectively.
Fig. 5 demonstrates the average identification time for all tags and the total number of tags is observed in the average identification time ratio between priority classes 2 and 1. However, when the values of $N_1$ and $N_2$ become wider than the previous case with $N_1=4$ and $N_2=12$, the index value of $C_1$ has more of an impact on the probability of collision than the increment of $C_2$.

Fig. 6 illustrates the relationship between the probability of collision and the values of $C_1$ and $C_2$. It can be noticed that the probability of collision rises as $C_1$ and $C_2$ increase. In case of $N_2 > N_1$, the increment of $C_2$ can cause more drastic effect on the probability of collision compared to the increment of $C_1$. In contrast, in case of $N_1 > N_2$, the increase of the value of $C_1$ has more of an impact on the probability of collision than the increment of $C_2$.

Fig. 7 shows the probability of idle as a function of $C_1$ and $C_2$. It can be seen that the probability of idle tends to decrease with the values of $C_1$ and $C_2$. In case of $N_2 > N_1$, the increment of $C_2$ can cause more effect on the probability of idle compared to the increment of $C_1$. On the other hand, in case of $N_1 > N_2$, the rise of $C_1$ has more impact on the probability of idle compared to the increase of $C_2$.

Figs. 8 and 9 illustrates the average identification time for priority class 1 and 2, respectively under different combinations of $N_1$ and $N_2$. It can be noticed that the average identification time for priority class 1 tends to increase with the values of $C_2$, whereas the average identification time for priority class 2 tends to increase with the values of $C_1$.

When the ratios of $N_1$ and $N_2$ increase, similar results to that of $N_1=4$ and $N_2=12$ are observed. However, the ranges of the average identification time for priority class 1 become narrower than the previous case with $N_1=4$ and $N_2=12$. On the other hand, the ranges of the average identification time for priority class 2 become wider than the previous case with $N_1=4$ and $N_2=12$.

Fig. 10 displays the average identification time ratio between priority classes 2 and 1 ($R_{21}$) as a function of $C_1$ and $C_2$. The results for the average identification time ratio between priority classes 2 and 1 are similar to those of the average identification time ratio between priority classes 2 and 1. However, when the ratios of $N_1$ and $N_2$ are increased, no significant changes are observed in the average identification time ratio between priority classes 2 and 1.

Fig. 11 shows the relationship between the average identification time for all tags and $R_{21}$. We can see from the results that the DC algorithm has weakness in controlling $R_{21}$, as there are a limited number of feasible values of $R_{21}$. In case of $N_1=4$, $N_2=12$, the values of $R_{21}$ lie between 0.8 and 1.4. Consider Fig. 11(b) where $N_1=8$, $N_2=8$, it can be seen that the values of $R_{21}$ are limited to the range between 0.76 and 1.32. Fig. 11(c) reveals that when $N_1=12$, $N_2=4$, the values of $R_{21}$ vary between 0.7 and 1.26.

B. Performance of DQ algorithm

Fig. 12 shows the average identification time for all tags as a function of $Q_1$ and $Q_2$ under different combinations of $N_1$ and $N_2$ as the values of $Q_1$ and $Q_2$ vary from 1 to 15. It can be seen that the average identification time for all tags tends
to increase with the increment of \( Q_1 \) and \( Q_2 \). When both \( Q_1 \) and \( Q_2 \) increase up to 15, the maximum average identification time for all tags is reached. The performance degradation is due to an excessive number of idle slots, as evident in Fig. 14.

Moreover, it is found that, at small values of \( Q_1 \) and \( Q_2 \) the average identification time for all tags decreases with the values of \( Q_1 \) and \( Q_2 \). This is simply because when the values of \( Q_1 \) and \( Q_2 \) are small, the number of all tags is relatively much higher than the number of available slots. In this scenario, collision will most likely be difficult to avoid. Therefore, an increase in the values of \( Q_1 \) and \( Q_2 \) can help reduce the number of collision slots and thus improving the system performance. When \( Q_1 \) and \( Q_2 \) increase up to a certain value, the minimum average identification time for all tags is attained and the values of \( Q_1 \) and \( Q_2 \) at this point will be referred to as the appropriate values of \( Q_1 \) and \( Q_2 \). When \( Q_1 \) and \( Q_2 \) further increases, the average identification time for all tags begins to increase and eventually reaches the maximum average identification time for all tags when \( Q_1 \) and \( Q_2 = 15 \). Table 1 summarizes the appropriate values of \( Q_1 \) and \( Q_2 \) for various combinations of \( N_1 \) and \( N_2 \). It is very interesting to see that the appropriate values of \( Q_1 \) and \( Q_2 \) for all different combinations of \( N_1 \) and \( N_2 \) are equal to 4 and 4, respectively.

Fig. 13 demonstrates the probability of collision as a function of \( Q_1 \) and \( Q_2 \). The probability of collision tends to decrease with the increment of \( Q_1 \) and \( Q_2 \). This is as expected.
because when the initial frame lengths for priority classes 1 and 2 increase, a lot of time the slots are idle. This results in the reduction of the probability of collision. Moreover, the maximum probability of collision is obtained when the value of $Q_1$ and $Q_2$ are both equal to 1. This is not surprising because when $Q_1$ and $Q_2$ are both equal to 1, the initial frame lengths for priority classes 1 and 2 are both equal to 2. In this scenario, collisions are difficult to avoid because the number of priority class 1 and 2 tags are relatively much higher than the initial frame lengths.

Table 1 shows the appropriate values of $Q_1$ and $Q_2$. It can be noticed that the minimum probability of idle is reached when the values of $Q_1$ and $Q_2$ are both equal to 1. This is as expected because when the initial frame lengths for priority classes 1 and 2
are both equal to 2, all tags contend against each other in the contention slots and suffer from collisions. Furthermore, it can be seen that the probability of idle tends to increase as $Q_1$ and $Q_2$ increase. This is because when the initial frame lengths for priority classes 1 and 2 are large, the tags do not access the slots often enough and results in the increment of the probability of idle.

Fig. 15 displays the average identification time for priority class 1 under different combinations of $N_1$ and $N_2$. It can be noticed that the average identification time of priority class 1 increases with the value of $Q_1$. This is because the increment of the initial frame lengths for priority class 1 will increase the number of idle slots and result in the increase of the average identification time for priority class 1 tags.

In addition, the average identification time of priority class 1 decreases with the value of $Q_2$. This is because the increment of the value of $Q_2$ will reduce the number of accesses from priority class 2 tags in the early slots and results in the increase of the probability of success of priority class 1. Similar results are shown in Fig. 16. In this figure, the average identification time of priority class 2 increases with the value of $Q_2$ and decreases with the value of $Q_1$.

Consider Fig. 17 that shows the average identification time ratio between priority classes 2 and 1 using different values of $Q_1$ and $Q_2$. As we can see, the average identification time ratio
between priority classes 2 and 1 increases with the value of $Q_2$. This is because when $Q_2$ is large, the priority class 2 tags do not access the slots frequently enough in the early slots. This results in the increment of the average identification time of priority class 2 and the decrement of the average identification time of priority class 1. In case of $N_1=4$, $N_2=12$, the maximum average identification time ratio between priority classes 2 and 1 is reached with $Q_1=2$ and $Q_2=15$. When the combinations of $N_1$ and $N_2$ are equal to $N_1=8$, $N_2=8$ and $N_1=12$, $N_2=4$, the maximum average identification time ratios between priority classes 2 and 1 are reached with $Q_1=3$ and $Q_2=15$. In addition, the ranges of the average identification time ratio between priority classes 2 and 1 become narrower than the previous case with $N_1=4$ and $N_2=12$.

Fig. 18 shows the relationship between the average identification time for all tags and $R_{21}$. We can see from the results that there are many different possible values of $R_{21}$. However, $DQ$ algorithm cannot control the values of $R_{21}$ to obtain the desired integer values. Furthermore, it can be noticed that the values of $Q_1$ and $Q_2$ are important parameters affecting the system performance. Therefore, care must be taken in choosing the values of $Q_1$ and $Q_2$.
C. Performance of DCQ algorithm

In the DCQ algorithm, the values of $C_1$, $C_2$, $Q_1$ and $Q_2$ can be adjusted simultaneously in order to achieve the desired $R_{21}$ while maintaining low delay. Fig. 19 displays the relationship between the average identification time for all tags and $R_{21}$. Note that in this figure, only the limited number of feasible integer values of $R_{21}$ are plotted. These feasible integer values are obtained by using the appropriate values of $C_1$, $C_2$, $Q_1$ and $Q_2$ in order to achieve the desired integer values of $R_{21}$ while maintaining low delay. The appropriate values of $C_1$, $C_2$, $Q_1$ and $Q_2$ for DCQ algorithm are illustrated in Figs. 20 and 21, respectively.

As can be seen from Fig. 19, the average identification time for all tags tends to rise as $R_{21}$ increases. This can be explained as follows. In order to achieve high $R_{21}$, the system has to increase the average identification time of priority class 2 tags. This results in high average identification time for all tags. Moreover, it can be observed that when the ratios of $N_1$ and $N_2$ increase, the ranges of $R_{21}$ become narrower than the case with $N_1=4$ and $N_2=12$.

D. Performance comparison of the proposed algorithms and the existing known algorithm

In this section, we compare the performance of all proposed algorithms and the existing known algorithm namely $Q$ algorithm, as illustrated in Fig. 22. In the $Q$ algorithm, all tags have the same priority and the system parameter settings of the $Q$ algorithm are given in the Table 2. Note that the results of the DCQ algorithm are obtained by using the appropriate system parameters. As we can see, at $R_{21} = 1$, no difference between two classes, the DC, DQ and DCQ algorithms give the same result as the $Q$ algorithm and the minimum average identification time for all tags of 23.73 can be reached. However, when $R_{21} > 1$, it can be observed that the identification time for all tags increases as $R_{21}$ increases. This is because the system has to limit the success rate of service class 2 tags in order to obtain the desired high $R_{21}$ and results in overall performance degradation. It is important to note that we are not interested in the case when $R_{21} < 1$ because we have defined the priority class 1 tags to have higher priority than the priority class 2 tags.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1$</td>
<td>0.1</td>
</tr>
<tr>
<td>$C_2$</td>
<td>0.1</td>
</tr>
<tr>
<td>Initial $Q_{fp}$</td>
<td>4.0</td>
</tr>
<tr>
<td>Minimum $Q_{fp}$</td>
<td>0.0</td>
</tr>
<tr>
<td>Maximum $Q_{fp}$</td>
<td>15.0</td>
</tr>
</tbody>
</table>

In addition, it can be noticed that the DCQ algorithm offers relatively superior performance. This is because in the DCQ algorithm, it is possible to adjust the values of $C_1$, $C_2$, $Q_1$ and $Q_2$ simultaneously. On the contrary, in the DC algorithm, we...
Fig. 19. Average identification time for all tags vs $R_{21}$ for the DCQ algorithm: (a) $N_1=4$, $N_2=12$ (b) $N_1=8$, $N_2=8$ (c) $N_1=12$, $N_2=4$.

Fig. 20. The appropriate values of $C_1$ and $C_2$ vs $R_{21}$ for the DCQ algorithm: (a) $N_1=4$, $N_2=12$ (b) $N_1=8$, $N_2=8$ (c) $N_1=12$, $N_2=4$.

Fig. 21. The appropriate values of $Q_1$ and $Q_2$ vs $R_{21}$ for the DCQ algorithm: (a) $N_1=4$, $N_2=12$ (b) $N_1=8$, $N_2=8$ (c) $N_1=12$, $N_2=4$.

can adjust only the values of $C_1$ and $C_2$ whereas in the DQ algorithm, only $Q_1$ and $Q_2$ can be adjusted. For these reasons, the DCQ algorithm is more flexible and hence potentially leading to more effective. However, when the ratios of $N_1$ and $N_2$ are increased, the ranges of $R_{21}$ become narrower than the case with $N_1=4$ and $N_2=12$.

IV. CONCLUSION

In this paper, we have presented three new anti-collision algorithms for RFID system with different delay requirements. Through the simulation results, we found that the minimum average identification time for all tags can be reached when there is no difference between two priority classes. In this case, the DC, DQ and DCQ algorithms become the Q algorithm. Moreover, we can conclude that the values of $C_1$, $C_2$, $Q_1$ and $Q_2$ are the important parameters that must be set appropriately for different system loads, so that the system can control the delay differentiation level and yet maintain high system performance. When comparing between all proposed algorithms, we found that the DCQ algorithm is more effective and flexible in adjusting the system parameters to meet the delay requirement than the DC and DQ algorithms. This is an essential step toward the design of anti-collision algorithms that support the
RFID system with different delay requirements.

ACKNOWLEDGMENT

The author would like to thank Thai-Nichi Institute of Technology for financial support of this work.

REFERENCES


AUTHOR PROFILE

Warakorn Srichavengsup obtained the B.Eng., M.Eng. and Ph.D. degree in Electrical Engineering from Chulalongkorn University, Bangkok, Thailand, in 1998, 2003 and 2009, respectively. He is currently a lecturer with the Department of Computer Engineering at Faculty of Engineering, Thai-Nichi Institute of Technology (TNI), Bangkok, Thailand. Prior to joining TNI, he was a visiting research student during 2008 with the Laboratory for Information and Decision Systems (LIDS) at the Massachusetts Institute of Technology (MIT). His main research interests are MAC protocol for high speed wireless local area networks, computer cryptography and information security.