Research of Matrix-based Grouping Method on Anti-collision Algorithm for RFID Tag Identification

Lvqing Yang, Caili Wang, Wenhua Zeng

1Software School of Xiamen University and Business Administration postdoctoral in Xiamen University, China, Email: lqyang@xmu.edu.cn
2Software School of Xiamen University, China, E-mail: franst1128@163.com
3Software School of Xiamen University, China, E-mail: whzeng@xmu.edu.cn

Abstract

Currently, the Binary Search Tree (BST) algorithm is mainly used for continuous identification signal processing on active RFID tags. However, tag population is unlimited increasing. With using the BST, the number of identified tags is limited at an average period of time. In this paper, we propose the BST algorithm of the matrix-based grouping to accelerate the recognition speed of tags. It is shown that the optimization algorithm is effective and practical by the theoretical research and simulation experiment. Compared with the BST algorithm, this method can effectively reduce the recognition time and improve the recognition speed of tags.

Keywords: The matrix-based grouping, Binary search tree, Anti-collision algorithm, RFID

1. Introduction

Radio Frequency Identification (RFID) is a new important automatic identification technology [1]. A RFID system overcomes challenges of other identification systems including bar code systems, smart cards, and biometrics (voice, fingerprinting, retina scanning) [2]. Because it does not require line-of-sight communication, sustains certain harsh physical environments, maintains a cost-low and power-efficient operation, and allows for simultaneous tag identification. In other words, a simple RFID system can send all kinds of information of things to the mobile network nodes. These nodes can be tracked and monitored, and trigger action or response to an action request [3].

A RFID system consists of radio tags, readers and the data processing system [4]. The electronic tags mainly include passive tags and active tags. The difference between the two tags is that whether tags actively make the transmission of signals. Without providing energy by readers, active tags are started by internal batteries that supply power. It usually leads to a longer distance with tens of meters or even hundreds of meters [5]. Therefore, active tags have been widely used in the personnel positioning and tracking in wireless sensor networks [6]. Whatever kinds of tags, tag collision is increasing while the number of tags is increasing. A large number of collisions result in slowing down the tag interrogation procedure. So the problem of anti-collision stands out [7].

Due to long interrogation procedure and high energy consumption of tags in the BST algorithm, the BST algorithm of the matrix-based grouping (BSTM) is proposed. With the BSTM, tags are grouped according to the number of tags that is identified and estimated by readers. The goal is to reduce collisions and searches while readers are recognizing tags. Then it can reduce recognition time and improve the searching efficiency of readers.

2. The theory of anti-collision and Binary Search Tree algorithm

2.1. The theory of anti-collision

For a specific RFID system, an arbitrary RFID tag owns the conviction of an Electronic Product Code (EPC). As multiple tags are sending EPCs in the scope of the reader rf (radio frequency) field, there are two or more tags that respond to a reader at the same time. That will produce a conflict that is referred to as a collision. The algorithm of solving the problem of collision is called anti-collision algorithm [8]. Of course, with multiple EPCs in the scope of the reader rf field, the data communication
between them can also cause data interference. But we rarely consider the situation. Anti-collision algorithm is to identify tags in turn by providing the corresponding strategy. At the same time, to prevent the appearance of the collision, a RFID system will take the corresponding technical measures to solve the problem of collision. These measures are referred to as anti-collision protocols. Anti-collision protocols can be realized by related commands of anti-collision algorithm [9].

2.2. BST(Binary Search Tree) algorithm

Among algorithms to solve the problem of anti-collision, BST is used flexibly and widely. To implement BST algorithm, it is necessary to identify the precise location of bits of data collision in readers. It makes interaction between a reader and many tags to get a selected tag from a large set of tags. In order to realize the BST algorithm, it usually takes the Manchester encoding to identify collision effectively. Logic "0" is encoded as a rising edge, and logic "1" is encoded as a falling edge [10]. It is shown in figure 1.

![Figure 1. The schematic diagram of the Manchester encoding](image)

Based on random numbers, the BST is used with the recursive way of working. The BT (Binary Tree) branch is made as a collision occurred. These branches are called two subsets [11]. These branches are getting smaller and smaller, until the branch only owns one information packet or is empty. Like the toss of a coin, the method of branches divides these packets into two branches randomly. In the first branch, the packet is "throwing positive" (value 0). Within the following time slot, it is to mainly solve collisions between the packets. If a collision occurs again, the first branch continues to be divided into two branches randomly. The process is repeated until a slot is empty or it makes the successful completion of a data transfer, then it returns to the previous branch. This process follows the principle of "First-In Last-Out". After all of the packets are successfully transmitted in the first branch, it is turn to transmit the second branch that is "throwing negative" (value 1) [12].

In conclusion, with the increase of the number of tags, not only the number of collisions but also the depth of BST will be increasing. It also needs to make multiple Request-Returns for information packets to identify a tag. So it was mentioned to change a binary tree into a quad-tree in Ref. [13]. And the author proposed a way of grouping proof, which uses the dynamic binary tree anti-collision algorithm to subgroup the tags in Ref. [14]. By these methods, the depth of the BST can be reduced. But with the increase of tags, the searching depth still increases relatively. So we take the method of the matrix-based grouping to reduce collisions and the depth of the BST. After multiple searches for the reader, the recognition speed of tags is up.

3. The BST algorithm of the matrix-based grouping

By the matrix-based grouping, the main goal of BSTM is to reduce the number of the identified tags during the tag interrogation procedure. In the following, the BSTM is described in detail by the principle and the procedure.

3.1. Algorithm principles

We assume that there are N tags in the scope of the reader rf field. So the number of nodes is “2N-1” in the BT that is formed by N tags, and the average depth of leaf nodes is:

![Diagram](image)
Therefore, the number of the average search times is \([\log_2 N] + 1\) to identify Tag-N from N tags. The time complexity is \(o(N \log_2 N)\) after all of N tags are identified \([15]\). It can be seen that the number of the average search times is becoming large when the value of N increases.

To identify all of N tags by using the basic BST algorithm, the total number of identification is:

\[
S_{BST} = \log_2 N + 1 + \log_2 (N - 1) + 1 + \ldots + \log_2 1 + 1 = \sum_{i=1}^{N} (\log_2 i + 1)
\]

(2)

The purpose of grouping is to reduce the search time of identifying tags by using the BST algorithm. In result, it performs multiple searches to continuously identify the tags. There needs to be pointed out that the number of tags of each group is not as little as possible while grouping. We should make reasonable grouping according to the current number of tags. Therefore, we have proposed the idea of a matrix-based tag ID grouping.

For the BSTM algorithm, the key is how to reasonably group tags. The method is to calculate the value of square root according to the number of tags. Then it is selected as the row value of the matrix with the smallest integer which is greater than the value of square root, that is:

\[
\text{Line} = \lceil \sqrt{N} \rceil
\]

(3)

On this basis, we calculate the column value of the matrix, that is:

\[
\text{Column} = \lfloor N / \lceil \sqrt{N} \rceil \rfloor
\]

(4)

The time which the BSTM algorithm spends includes two aspects in theory:

1) The time of the matrix-based grouping, that is:

\[
S_1 = \lceil \sqrt{N} \rceil \times \lfloor N / \lceil \sqrt{N} \rceil \rfloor + o(1)
\]

(5)

2) The time of searching for the BST algorithm, that is:

\[
S_2 = \lceil \sqrt{N} \rceil \times \sum_{i=1}^{\lfloor N / \lceil \sqrt{N} \rceil \rfloor} (\log_2 i + 1) + \sum_{i=1}^{\lfloor N / \lceil \sqrt{N} \rceil \rfloor} (\log_2 i + 1)
\]

(6)

So, the total number of identification is:

\[
S_{BSTM} = S_1 + S_2
\]

(7)

Where the symbol of “\(N\)” means the number of tags; the symbol of “\(\lfloor \rfloor\)” means the value that is the minimum integer which is greater than a certain value; the symbol of “\(\lceil \rceil\)” means the value that is the maximum integer which is less than a certain value.

According to (2) and (5), relevant data for the average search times is listed for the BST algorithm and the BSTM algorithm. It is shown in table 1.

<table>
<thead>
<tr>
<th>Number of tags</th>
<th>BST</th>
<th>BSTM</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>38</td>
<td>23</td>
</tr>
<tr>
<td>24</td>
<td>103</td>
<td>71</td>
</tr>
<tr>
<td>56</td>
<td>305</td>
<td>211</td>
</tr>
<tr>
<td>61</td>
<td>340</td>
<td>223</td>
</tr>
<tr>
<td>81</td>
<td>483</td>
<td>329</td>
</tr>
<tr>
<td>100</td>
<td>625</td>
<td>418</td>
</tr>
<tr>
<td>150</td>
<td>1023</td>
<td>634</td>
</tr>
<tr>
<td>200</td>
<td>1446</td>
<td>890</td>
</tr>
</tbody>
</table>

Table 1. Compare of the search time before and after improved algorithm
It shows that the search time is significantly reduced in theory in table 1.

3.2. Algorithm procedures

With the BSTM algorithm, tags are grouped into an approximate square matrix according to ID. Then it makes identification to select a row number of tags. It is shown in figure 2.

\[
\begin{bmatrix}
tag_1 & \ntag_2 \\
tag_3 & \ntag_4 \\
\end{bmatrix}
\]

(a) \(N = 4\)

\[
\begin{bmatrix}
tag_1 & \ntag_2 & \ntag_3 \\
tag_4 & \ntag_5 & \ntag_6 \\
tag_7 & \ntag_8 & \ntag_9 \\
\end{bmatrix}
\]

(b) \(N = 9\)

\[
\begin{bmatrix}
tag_1 & \ntag_2 & \ntag_3 & \ntag_4 \\
tag_5 & \ntag_6 & \ntag_7 & \ntag_8 \\
tag_9 & \ntag_{10} & \ntag_{11} & \ntag_{12} \\
tag_{13} & \ntag_{14} & \ntag_{15} & (empty) \\
\end{bmatrix}
\]

(c) \(N = 15\)

**Figure 2.** Examples of \(N\) tags with the matrix-based grouping

As the reader can’t directly access to the number of tags in the scope of the reader rf field. With the first collision detection, the way to estimate the number of tags is chosen according to the number of collision bits [16].

The basic steps of the estimation command are as follows:

The estimation command is executed according to the fixed-length of the requested collision-bits. Then count the number of the collision-bits;

Based on the number of collision-bits, the number of tags is estimated within the scope of the reader.

The algorithm flow of the BSTM is shown in the figure 3.
Figure 3. The flowcharting of the BSTM algorithm
The basic steps of the algorithm flow are following:

1) System initialization. The serial number of each tag is a fixed-length string that consists of bits. The value of each bit is 0 or 1.
2) The reader sends the Request0 command to the tag to request the IDs of tags. Then the reader receives IDs which tags return.
3) According to the codes, the reader detects collision bits.
4) Do the estimating command. With the number of the collision bits, the number of tags which the reader estimates is \( N \).
5) Based on \( N \), the reader calculates that each group should include \( M \) tags in the matrix.
6) The reader sends the command of Request to any \( M \) tags. Tags return the IDs to the reader. Then the reader identifies \( M \) tags by multiple Request-Returns.
7) After identifying a group of tags, the remaining number of tags is "flag". The value of flag is passed to \( N \). By judging the value of \( N \), the reader decides whether or not to proceed with the identification. The process is repeated. It doesn’t stop until all tags are identified in the scope of the reader rf field.

4. Algorithm simulation and performance analysis

We make algorithm simulation in the Matlab software via computer. While simulating, we don’t temporarily consider the reader’s control, redundancy check, BER (Bit Error Ratio) and so on. The length of ID is 16-bit. The average of the results of 50 experiments is selected as the result.

In this paper, two indicators are selected to analyze the performance for the improvement before and after the collision algorithm. The first is the time to identify all of tags. The second is the throughput per unit time in the reader. In the simulation experiment, the time is shown as the time slot that includes query time slot and response time slot; the throughput is shown as the number of identified tags per unit time slot, that is:

\[
\text{Throughput} = \frac{N}{\text{time slot}}
\]  

(8)

The experimental results are shown in figure 4 and 5.

![Figure 4](image-url)

**Figure 4.** Compare with the time slot. BST denotes Binary Search Tree; BSTM denotes Binary Search Tree of the Matrix-based grouping; QT denotes Quad-tree; QTM denotes Quad-tree of the Matrix-based grouping
While the value of N is increasing, the time slot is decreasing and the throughput per unit time slot is increasing correspondingly, as is shown in figure 4 and figure 5. It proves that the BSTM algorithm is helpful to improve the operating efficiency of the reader. In conclusion, the BSTM is better advantages than the BST.

In addition, the idea of the matrix-based grouping is independent without the BST algorithm. On the basis, the idea of grouping is applied into the quad-tree anti-collision algorithm. The performance is shown in figure 4 and figure 5 that the efficiency of the QST algorithm gets also improved.

5. Conclusion

In this paper, we make an analysis about the RFID anti-collision. On this basis, the BST algorithm is researched in details. Then the correlated algorithm has been improved and verified by simulation experiments. The result shows that it not only reduces the number of the time slot, but also improves the throughput of the system by the matrix-based grouping. The purpose is achieved to accelerate the recognition speed of tags. So the idea of the matrix-based grouping owns its advantages. Meanwhile the idea can be applied into the QST algorithm. It also shows a good performance.

Future research directions include two aspects. Firstly, on the foundation of the experimental achievement, the matrix-based grouping will be tried to apply into other different types of anti-collision algorithms and verify their effects. Secondly, we will probe into the method of the multidimensional grouping that is based on the matrix-based grouping. The purpose is to continually deepen the research on the algorithm and enlarge its scope of applications.

6. References


