AN EFFICIENT REDUNDANT INFORMATION COLLECTION PROTOCOL FOR THE GROUP INTEGRITY OF TAGS IN RFID SYSTEMS

Reem Alkanhel^{1, 2*} and Marcel Ambroze²

¹*Department of Information Technology, Princess Nourah bint Abdulrahman University, Saudi Arabia*

²*Department of Computing, Electronics and Mathematics, Plymouth University, United Kingdom*

¹rialkanhal@pnu.edu.sa, ²marcel.ambroze@plymouth.ac.uk

Abstract— Verifying the integrity of groups of radio frequency identification (RFID) tagged objects or recovering the identifiers of lost objects is essential in RFID systems. Existing autonomous verification methods write redundant data, which reference other tags in the group, into the memory of tags. This study aimed to efficiently collect such redundant information from a group of tags. Existing possible solutions are not efficient in solving the problem, since they do not distinguish this redundancy among tags, which results in long collection times. To achieve time efficiency, a new protocol called the Redundant Information Collection (RIC) protocol is proposed. It is based on a Bloom filter that efficiently filters redundant tag information. Simulation results are presented showing that the collection time of the proposed protocol is around four times the lower bound. The proposed protocol also outperforms existing possible solutions by saving nearly 37%–84% of the collection time.

Keywords— Tag information collection, redundant information, Bloom filter, filtering, RFID system, time efficiency

1. INTRODUCTION

Radio frequency identification (RFID) is an automatic technology that enables the identification and tracking of objects through radio waves. A typical RFID system consists of the following three main components: RFID tags, readers and back-end database servers [1]. Verifying the integrity of a group of tagged objects or recovering the identifiers (IDs) of lost objects is essential in various activities to find lost tagged objects. Autonomous verification methods have been developed to increase the accuracy of determining the number of lost tag IDs or recovering all lost tag IDs without the need for external systems [2]–[7]. This is accomplished by treating a group of n tag IDs denoted by $ID_1, ID_2, ID_3, \dots, ID_n$ as packet symbols and by encoding and decoding them in a way similar to that of binary erasure channels. An $m \times n$ parity check matrix of low density parity check codes is used as a group generator matrix to produce m parity check values as follows:

Received: October 20, 2019

Reviewed: December 30, 2019

Accepted: January 14, 2020

*Corresponding Author

$$\begin{Bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{Bmatrix} = H \cdot \begin{Bmatrix} ID_1 \\ ID_2 \\ \vdots \\ ID_n \end{Bmatrix} \quad (1)$$

where, H is the parity check matrix and $c_i, i \in \{1..m\}$, is the parity check value of the i -th row.

Previous research [8], [9] on the use of the progressive-edge-growth-based (PEG-based) method has shown that it achieves considerable missing recovery improvements compared with other extended grouping methods in [4]–[7] with a low decoding complexity. As a requirement for those autonomous methods, redundant data are written into the memory of RFID tags, which reference other tags in the group. They are required to build Tanner graphs during the decoding process and thus identify lost tag IDs. Each tag records all parity check values (c_i) in which they are involved, as well as the short IDs (s_i) of other tags belonging to the same row of the parity check matrix. Tags of the same row maintain the same set of information. For example, tag_1 is involved in the parity check value c_1 with tag_4 and tag_5 . Therefore, tag_1 records c_1 as well as s_4 and s_5 as depicted in Fig. 1.

For the decoding process, the reader needs to collect the parity check values and their short IDs from available tags. Apparently, designing an efficient collection protocol is needed to read such redundant information from the tags in RFID systems. The top performance standard for the protocol is time efficiency, because RFID systems have consecutive operations to perform, such as identifying lost tag IDs or determining the number of lost tag IDs. When the time is long, the task of collecting tag information may delay these scheduled processes of the system, which is undesirable, especially in large-RFID environments. Similar research on collecting tag information has been conducted before by [10]–[14]. However, these studies required the collection of all tags' information. To the best of our knowledge, no prior work has focused on how to efficiently filter and collect redundant tag information. The goal of the present study, therefore, was to design an efficient protocol for collecting redundant tag information in approximately the optimal collection time.

Fig. 1 An example of the information stored in RFID tags belonging to a group of six tags

In this study, three existing solutions to general and different problems that could be applied to collect redundant tag information were examined, and these are simply called possible solutions. None of them are efficient, since they do not distinguish redundancy among tags, which results in long collection times. This issue has motivated the development of the proposed protocol, called the Redundant Information Collection

(RIC) protocol. It is based on a Bloom filter (BF) that efficiently filters the redundant tag information, thereby considerably decreasing the communication cost and, consequently, the collection time. Simulation results are presented showing that the proposed protocol outperforms existing solutions by saving around 37%–84% of the collection time, which is nearly four times the lower bound. This characteristic makes the RIC protocol a promising candidate for collecting redundant tag information in the group integrity of tags in RFID systems and other similar systems.

The remainder of this paper is organised as follows. Related work is reviewed in Section 2. Section 3 presents the preliminaries of this study, and Section 4 discusses the possible solutions. Section 5 provides a detailed description of the proposed protocol, and Section 6 evaluates the performance of the proposed protocol with simulations. Finally, Section 7 concludes the study.

2. RELATED WORK

Many existing RFID studies mainly focus on proposing ID collection protocols that collect tag IDs in the interrogation zone of the reader. These protocols are mainly classified into ALOHA-based [15], [16] and tree-based [17], [18] protocols. In ALOHA-based protocols, the reader sends a request message to the tags. Then, each tag transmits its ID to the reader at randomly chosen time slots. Consequently, more than one tag may possibly transmit their IDs at the same time slot; in this case, the time slot becomes a collision slot. On the other hand, if only one tag transmits its ID at a certain time, the tag is successfully identified. In tree-based protocols, collisions are reduced by building tree traversal approaches. When a collision occurs, the reader divides the tags into two subgroups and requests the subgroups with a smaller number of tags. The reader continues to divide the tags until all the tags are identified. Whilst the ID collection protocol can be used to collect tag information, the transmission overhead and time delay could be excessively high for redundant tag information (explained in Section 4).

The studies that are most related to this work are [10]–[14]. The authors in [10] proposed the Single-hash Information Collection (SIC) protocol and the improved version, the Multi-hash Information Collection (MIC) protocol, to periodically collect information from sensor-based RFID systems, where tags are combined with miniaturised sensors, with a minimum collection time. In such systems, RFID tags could provide information about the tagged items or their environments. By utilising several hash functions, the MIC protocol can resolve most hash collisions in a frame and thus significantly enhance the protocol performance. References [11], [12] studied the same problem but with respect to energy efficiency for active tags. The tag ordering polling protocol and the improved protocol were presented to read information from tags with the smallest amount of energy consumption. A follow-up work by [13] complemented [10] with an additional protocol that is primarily designed to read sensor-produced information from RFID tags with multiple readers. The extension study in [14] targeted an unknown subset of tags with one reader. The proposed protocols utilised BFs that are distributively built and transferred from the tags to the reader to identify the target tags efficiently. The reader can then build a hash vector from a frame of time slots to link each identified tag to each slot in order to send its information.

Many methods in the literature filter out the duplicate information in RFID systems [19], [20]. Duplicate data are produced because an RFID reader reads the same tag multiple times, or the same tag is read by multiple readers. All these methods concentrated on filtering duplicate information at the reader's or the middleware's side before the information is transmitted to back-end database servers for processing, which causes communication overheads between the reader and the tags. The majority of the proposed methods are based on BFs. Generally, the received data are maintained in BFs to eliminate duplicates and thus reduce memory and communication consumptions.

3. PRELIMINARIES

3.1. SYSTEM MODEL AND PROBLEM STATEMENT

Consider an RFID system with one reader and many passive tags. On the basis of the EPC global Class-1 Generation-2 standard [21], each tag stores the 96-bit ID that uniquely identifies the tagged item. Tags can perform some calculations, such as hashing. The communication between the reader and the tags uses a framed, slotted Aloha protocol, which is known as the reader-talks-first (RTF) mode. The signal from the reader synchronises the clocks of the tags. In each cycle of communication, a reader broadcasts a request in a time slot, and then multiple tags reply in the following slotted frame. Each tag will randomly choose one slot. If no tag sends a signal in a slot, the slot is called an empty slot. If one or more tags send signals in one slot, the slot is known to be a non-empty slot. The non-empty slot can also be classified into a singleton slot when only one tag sends a signal; in this case, the transmitted signal can be successfully decoded by the reader. On the other hand, when two or more tags respond in the same slot, the signals collide and thus, the reader cannot correctly decode the signals. The non-empty slot here becomes a collision slot. In the collision slot, if all tags send identical data simultaneously, the data are successfully received and interpreted by the reader [22]. As stated in the EPC standard, the transmission rate between the reader and the tags is asymmetric and subject to the environment.

Consider a set of RFID tag IDs denoted as $ID = \{ID_1, ID_2, ID_3, \dots, ID_n\}$ corresponding to a set of tags represented as $TAG = \{tag_1, tag_2, tag_3, \dots, tag_n\}$, where n is the number of tags in the group. A parity check matrix H of size $m \times n$, which is constructed using the PEG-based method [8], is used as a generator matrix to encode a group of tag IDs of size n and thus generate a set of parity check values of size m denoted by $C = \{c_1, c_2, c_3, \dots, c_m\}$. The parity check value c_i is computed as the bitwise exclusive or (XOR) of the IDs of the tags belonging to the same row of the parity check matrix. The bit length of each c_i is equal to the bit length of the tag ID, e.g. 96 bits. To recover the lost tag IDs, each tag needs to store its parity check values c_i , as well as a set of short IDs s_i , referring to other tags belonging to the same row of the parity check matrix. The bit length of each s_i is set to the minimum number of bits that are necessary to distinguish a group of tags; for example, a short ID of 8 bits is enough for a group comprising less than 256 tags. In this study, for the purpose of collecting redundant tag information, a group of tags belonging to the same row of the parity check matrix maintained the same order of short IDs. Therefore, the short ID of each tag needs to be recorded in its memory. For example, tag_1 , tag_4 , and tag_5 in Fig.1 belong to the first row of the parity check matrix, and thus these tags record the values of c_1 , s_1 , s_4 , and s_5 in the same order. How the set of c_i and s_i is stored into the tag's memory is outside the scope of this study.

The problem is to design a protocol for a reader to filter and collect redundant information from a group of tags in the interrogation zone of the reader with the minimum collection time. The redundant information collection problem is different from those studied before in sensor-based RFID systems [10]–[14]. In sensor-based RFID systems, the tags have no knowledge about the information the reader has and should send all their information. On the other hand, when collecting redundant information, each tag needs to be efficiently informed about the collected information in order not to send it again.

3.2. PERFORMANCE LOWER BOUND

Let t_{bit} denote the time needed to send one bit. In group integrity applications, the information related to one parity check value and its short IDs is denoted by $c_i + s_i \times l$ and can be expressed as x , where l represents the row weights of the parity check matrix used for the encoding. All the tags contain the same amounts of parity check information,

which are denoted by $j \times (c_i + s_i \times l)$ and can be expressed as $j \times x$, where j is the column weights of the parity check matrix. Therefore, each tag needs a time whose length is equal to $x \times t_{bit}$ in order to send one piece of parity check information x . For collecting all tag information, the lower bound on the collection time for a group of n tags is $n \times j \times x \times t_{bit}$. For collecting redundant information, the lower bound is much less than $n \times j \times x \times t_{bit}$. The reader needs to collect each c_i and its s_i only once. Thus, the lower bound on the collection time is $m \times x \times t_{bit}$. This lower bound is unlikely to be achieved since the reader requires extra time to send its request and coordinate the protocol process. Despite this, it provides a benchmark to assess the performance of the designed protocol.

4. POSSIBLE SOLUTIONS

In this section, three existing solutions to general and different problems that could be applied to collect redundant tag information are examined, and these are simply called possible solutions. None of them are efficient in solving the redundant information collection problem, since they do not distinguish redundancy among tags, which results in long collection times. This gives clear knowledge about the solution of the redundant tag information collection problem, which motivated the new protocol proposed in the next section.

4.1. BASELINE COLLECTION PROTOCOL

A baseline collection (BC) protocol for collecting tag information is to let the RFID reader send the ID of one tag. Then, it receives the information of the related tag. The reader repeats this process until it gets all the information needed [10]. For collecting redundant information, the tag does not have knowledge about the information that the reader has, so it must send its information. Thus, the total time to collect one piece of parity check information x from a tag is $(ID_i \times t_{bit} + x \times t_{bit})$, and the optimal total time for collecting all parity check information from a group is $m \times (ID_i \times t_{bit} + x \times t_{bit})$. The advantage of this protocol is that the communication between the reader and the tags is one-to-one, where collisions cannot occur. On the other hand, the reader needs to transmit the tag ID in order to collect related tag information, which increases communication costs because the tag ID is 96 bits. Consequently, the time is significantly above the lower bound value $m \times x \times t_{bit}$.

4.2. ID COLLECTION PROTOCOL

If one of the ALOHA-based ID collection protocols [15], [16] is borrowed to collect redundant tag information when it collects the tag IDs, the collection time is high because each tag has to send all its information. In addition, the protocol has to consider the entire tag group n each time when collecting the tag's information. In the enhanced version presented in [15], the optimal time for collecting n tag information is $e \times n \times j \times x \times t_{bit}$, where e is the natural logarithm constant and is equal to 2.72. This means that each tag resends its information, on average, 2.72 times, which is the theoretically ideal number. Experiments and simulation results demonstrated that the best performance of tree-based protocols [17], [18] is equivalent to that of the ALOHA-based protocols [10].

4.3. MULTI-HASH INFORMATION COLLECTION PROTOCOL

Reference [10] presented the SIC and MIC protocols for collecting tag information in sensor-based RFID systems. Only the MIC protocol is considered here because it is the improved protocol. The MIC protocol is performed stage by stage. In each stage, the reader links tags to time slots using k hash functions. Then, the reader uses a hash selection vector to notify the tags about the links. Each item in the vector is related to a time slot at the same index location. The reader sets the item to 0 if no tag is linked to a

slot. If a tag is linked to a slot with the z -th hash function, where $1 \leq z \leq k$, the reader sets the related item to z . The size of an item is $\lceil \log_2(k+1) \rceil$. After receiving the vector, each tag is required to check the items in the vector that it is linked to by using the k hash functions. Let T_z , $1 \leq z \leq k$, be the item that the tag is linked to by the z -th hash function. Each tag checks the items in the order from T_1 to T_k . If an item T_z has the value of z , the tag concludes that it will be allocated to a slot with the z -th hash function. The above process iterates stage after stage until all tags send their information. The first stage targets n tags for slot allocation, and the frame size is equal to n . Each successive stage targets a smaller number of tags, and thus its frame size is reduced. The expected total collection time of the MIC protocol is:

$$\frac{n}{32P_k} \times ID_i \times t_{bit} + \frac{n}{P_k} \times j \times x \times t_{bit} \quad (2)$$

where, P_k is the probability that a tag is linked to one time slot in every stage when the k hash function is used. It has been proven in [10] that when $k = 7$, the collection time reaches the optimal value of:

$$0.036 \times n \times ID_i \times t_{bit} + 1.16 \times n \times j \times x \times t_{bit} \quad (3)$$

where, $P_7 = 86.1\%$. Similar to the BC protocols, when the MIC is used for collecting redundant information, each tag does not have knowledge about the information that other tags have sent, so it must send its information. Thus, the optimal total time for collecting m parity check information from a group is:

$$0.036 \times m \times ID_i \times t_{bit} + 1.16 \times m \times x \times t_{bit} \quad (4)$$

5. REDUNDANT INFORMATION COLLECTION PROTOCOL

This section presents the RIC protocol, which is designed to collect and filter redundant tag information at the tag's side. The proposed protocol aims to avoid the redundant transmissions of c_i , as well as the corresponding s_i . The process of the RIC protocol consists of several rounds that are repeated until all c_i are read. In each round, the reader transmits a request, and then tags reply with their information in the successive slotted time frame. Each tag must check whether its information has been sent previously by other tags before it sends any information. The reader maintains a list of all collected parity check values and their short IDs. Reporting the collected parity check values to the tags in order to prevent transmitting redundant information in the subsequent rounds is too costly, especially when the set of tags in the group is large. To accomplish a high level of filtering efficiency, the collected parity check values are encoded into a BF, which is then sent instead of the actual parity check values. In other words, the BF is used as a tool to carry the aggregated parity check values and filter the tag's information at the tag's side.

A BF is a data structure that can store the items of a set in a space-efficient way and support membership requests [23]. A BF of a set $W = \{w_1, w_2, \dots, w_y\}$ containing y elements is described by a hashed Boolean vector of b bits, all initialised to 0. A BF uses k independent hash functions $h_1, \dots, h_k$. These hash functions map each element $w \in W$ to the bits $h_z(w) \in \{1, 2, \dots, b\}$ for $1 \leq z \leq k$, which are set to 1. For ideal performance, each of the k hash functions should be a part of the class of universal hash functions. In other words, it should hash each item to a uniform random number. In order to check whether an item $q \in W$, k hash functions are applied to q . Then, all $h_z(q)$ are checked to determine whether they are set to 1. Otherwise, $q \notin W$. Testing a membership by using a BF may yield a false positive. That is, it suggests that an item $q \in W$, although it is not. For many applications in practical situations, false positives may be acceptable as long as their rate

is adequately small [24]. From [23], the probability of a false positive or the false positive rate P_{FP} of the BF constructed from m elements in the RIC protocol can be represented as:

$$P_{FP} = (1 - (1 - \frac{1}{b})^{mk})^k \approx \left(1 - e^{-\frac{mk}{b}}\right)^k \quad (5)$$

The minimum value of P_{FP} can be found by taking the derivative of (5) with respect to k as $\frac{dP_{FP}}{dk} = 0$, which gives the optimal value of k as:

$$k = \ln 2 \times \frac{b}{m} \quad (6)$$

This results in the false positive probability of $(\frac{1}{2})^k = (\frac{1}{2})^{\ln 2 \frac{b}{m}}$. In the implementation of the RIC protocol, the number of parity check values m is known a priori. Given a desired P_{FP} and the number of elements m , the optimal length of the BF can be computed as:

$$b = -\frac{m \times \ln P_{FP}}{(\ln 2)^2} \quad (7)$$

For example, if the desired P_{FP} is 0.0001, signifying accuracy of 99.99%, the optimal length of the BF is around $19.2 \times m$ bits. Implementing k hash functions on tags may cause processing overheads because each tag needs to perform the calculations independently by using its limited resources [25]. However, the implementation of hash functions can be simplified to decrease the complexity of a tag's circuit; for example, a pre-stored logical ring of random numbers may be used to produce hash values, as described in [10].

5.1. PROTOCOL DESCRIPTION

The RIC protocol consists of several rounds. In each round of the communication, the reader sends a request message containing two parameters, r and f , where r is a random number and f is the frame size; that is, the number of slots in the frame. Upon receiving the request, each interrogated tag is randomly mapped to a slot in the frame at the index $h(r, c_i) \bmod f$, where $h(\cdot)$ is a hash function shared by all tags whose range is $[0, \dots, f-1]$. As each tag takes its c_i as the hash seed, the tags having the same parity check values c_i choose the same slot. Then, they send their identical parity check information x , and thus the signals can be decoded by the reader. In this case, the slot can be called a successful slot. On the other hand, some tags having different parity check values c_i may respond at the same slot, and the slot here is treated as a collision slot; therefore, the reader cannot decode the signals encoded from a set of un-identical parity check information. The reader stores the collected parity check values c_i into a list denoted as $D = \{d_1, d_2, d_3, \dots, d_a\}$. Then, it constructs a BF vector by mapping the collected parity check values into b bits array using k hash functions, as described in Algorithm 1. Given a desired false positive rate P_{FP} and the number of elements in D , the optimal value of hash functions and the optimal length of the BF constructed in the RIC protocol can be computed using (6) and (7), respectively.

The reader again sends another request to collect a new set of parity check values c_i . The reader also sends the constructed BF and corresponding parameters b and k . Before sending any parity check information, each tag must check whether its c_i has been sent previously by other tags; and here, the BF plays an important role by meeting this requirement. Each tag performs membership tests for all c_i it contains by using Algorithm 2. For each one, it performs k hash functions, which are identical to those used to construct the BF at the reader's side. It then tests whether the bit $h_z(c_i)$ is set to 1 for $1 \leq z$

$\leq k$. If so, the tag will treat the tested c_i as redundant information and will not send it to the reader. Otherwise, the tested c_i and all its s_i are sent to the reader. After receiving the tag information, the reader updates its list and reconstructs the BF. The process of filtering and collecting information will be repeated until all tags report their information. The processes involved in all rounds are identical, except that at the beginning of the first round, during which the list of the reader is empty, and thus b is equal to zero, which means that the BF is not sent. A parity check value c_i that is not sent by a tag due to the false positive probability can be handled by the BFs in the successive rounds. Therefore, the parity check value c_i that is not collected in a round will be collected in a later round. Algorithm 3 outlines the proposed protocol.

Algorithm 1: Constructing the BF

Input: D, P_{FP}

Output: b, k, BF

```

1: Calculate  $b$  and  $k$ 
2:  $BF$  = allocate  $b$  bits initialised to 0
3: for each  $d_i$  in  $D$  :
4:   for  $z = 1$  to  $k$  do
5:      $BF[h_z(d_i)] = 1$ 
6:   end for
7: end for each

```

Algorithm 2: Membership testing

Input: elm, BF, k

Output: Yes/No

```

1: for  $j = 1$  to  $k$  do
2:   if  $BF[h_j(elm)] \neq 1$  then
3:     return No
4:   end if
5: end for
6: return Yes

```

Algorithm 3: The RIC protocol

Input: P_{FP}

Output: D containing all c_i and its s_i in the interrogated region

```

1 :  $D = \phi$ 
2 : Repeat
3 :   Constructing BF ( $D, P_{FP}$ ) return ( $b, k, BF$ )
4 :   The reader sends  $\{r, f, b, k, BF\}$ 
5 :   for each  $c_i$  in  $tag_i$  :
6 :     membership testing ( $c_i, BF, k$ ) return Yes/ No
7 :     if No then
8 :        $tag_i$  sends its  $c_i$  and the corresponding  $s_i$  at index  $h(r, c_i)$ 
9 :        $D = D \cup c_i$  sent by  $tag_i$ 
10:      go to step (13)
11:    end if
12:  end for each
13: Until all tags send their information

```

5.2. PERFORMANCE ANALYSIS

In this section, the expected collection time of the RIC protocol and the optimal value of the frame size f are derived. The expected collection time T_i of the i -th round is

composed of the time needed by the reader to send the BF vector and the time needed by the tags to send the information. The reader broadcasts a request message, including r, f , the BF vector and its parameters. The transmissions from the reader, except the BF vector, are very small and thus can be ignored. Note that the BF vector may be too long, so it can be split into multiple segments of 96 bits, and each segment is sequentially transmitted, as in [10], [12]. Thus, the expected time to send b bits vector of the BF is calculated as:

$$\frac{b}{96} \times t_{bit} \quad (8)$$

From (7), b can be expressed as:

$$- \frac{|D| \times \ln P_{FP}}{(\ln 2)^2} \quad (9)$$

The time for the tags to test the set of parity check values from the BF is also ignored. Recall that one piece of parity check information is denoted by x ; the expected time for the tags to send their x information can be calculated as:

$$E_{succ} \times x \times t_{bit} \quad (10)$$

where, E_{succ} is the expected number of successful slots in the frame. Therefore, the expected collection time T_i of the i -th round is:

$$T_i = \frac{b_i}{96} \times t_{bit} + E_{succ,i} \times x \times t_{bit} \quad (11)$$

In the ID collection protocols [15], [16], when the reader uses a frame of size f and the number of replying tags is n , the probability that r tags reply in one slot is a binomial distribution and can be expressed as:

$$p(y = r) = \binom{n}{r} p^r (1 - p)^{n-r} \quad (12)$$

where, p is a success probability and y is a random variable. Because each slot in the frame has equal probability, (12) can be written as:

$$p(r) = \binom{n}{r} \left(\frac{1}{f}\right)^r \left(1 - \frac{1}{f}\right)^{n-r} \quad (13)$$

Thus, the expected number of read tags during one read cycle can be expressed as:

$$\lambda = f \binom{n}{1} \left(\frac{1}{f}\right) \left(1 - \frac{1}{f}\right)^{n-1} \quad (14)$$

In the RIC protocol, each tag uses its c_i to choose one slot, so the tags having the same c_i must share the same slot, which is called the successful slot. By treating this group of tags as a single tag, the successful slot can be seen as a singleton slot. Therefore, the expected number of successful slots can be calculated as:

$$\begin{aligned} E_{succ,i} &= f_i \binom{m_i}{1} \left(\frac{1}{f_i}\right) \left(1 - \frac{1}{f_i}\right)^{m_i-1} \\ &= m_i \left(1 - \frac{1}{f_i}\right)^{m_i-1} \\ &\approx m_i \cdot e^{-\frac{m_i-1}{f_i}} \approx m_i \cdot e^{-\frac{m_i}{f_i}} \quad \text{as } m_i, f_i \rightarrow \infty \end{aligned} \quad (15)$$

where, m_i is the number of x to be collected before the i -th round. By letting the first derivative of T_i be zero, the minimum value of T_i can be found as:

$$\frac{d(T_i)}{df_i} = \frac{d(E_{succ,i})}{df_i} \times x \times t_{bit} = 0 \quad (16)$$

where, $\frac{d(E_{succ,i})}{df_i}$ can be obtained from (15) as:

$$\frac{d(E_{succ,i})}{df_i} \approx \frac{m_i^2 e^{-\frac{m_i}{f_i}}}{f_i^2} \quad (17)$$

when f_i is large. Using Taylor series, (17) can be written as:

$$\frac{d(E_{succ,i})}{df_i} \approx \frac{m_i^2 (1 - \frac{m_i}{f_i})}{f_i^2} \quad (18)$$

By solving (16) numerically, the optimal value of f_i that minimises T_i is approximately equal to m_i .

5.3. CARDINALITY ESTIMATION

The reader can estimate the number of the target parity check values m before running the RIC protocol. To accomplish the estimation, the reader transmits a random number r and a frame size f . Each interrogated tag sends a short message at a time slot $h(r, c_i) \bmod f$. Some tags having the same c_i will respond at the same slot, whereas some tags having different parity check values c_i may respond at the same slot. According to the EPCglobal Class-1 Generation-2 standard [21], only one bit is required to distinguish an empty slot from a non-empty slot. For each received signal at the reader side, if the channel is idle, the received signal is set to 0. If the channel is busy, which indicates that at least one tag sends, the received signal is set to 1. The tag sends a short message of one bit at the corresponding time slot. After receiving the messages, the RFID reader checks the state of the slots. Therefore, by counting the number of empty slots, m can be estimated. The additional time overhead for estimating m of the interrogated tags is measured under various group sizes n . Table I demonstrates the estimated time of the RIC protocol, which is very small. In Section 6 below, it is shown that the RIC protocol outperforms other information collection protocols. Therefore, taking into account the estimation overhead will not affect the performance of the RIC protocol.

Table I. Estimation Time Comparison (in Seconds) Under Different Group Sizes

n	1000	2000	3000	5000
Estimation time	0.32	0.64	0.96	1.6

6. SIMULATION RESULTS

In this section, the performance of the RIC protocol is evaluated with Matlab. As no existing method aims to collect redundant tag information at the tag's side, three protocols, including the BC, MIC with seven hash functions and the enhanced version of the ALOHA-based (EA) protocols presented in [15], were implemented, and their collection times are measured and compared with those of the RIC protocol.

6.1. SIMULATION SETTINGS

The simulation settings follow the specifications of the EPCglobal Class-1 Generation-2 standard [21]. A waiting time of 302 μ s was used to separate any two successive communications between the reader and the tags. The transmission rate of a tag is 53 Kb/s. A tag needs 18.87 μ s to send one bit. The length of time a tag needs to send its information was obtained as the addition of the time for sending the information and a waiting time. For instance, if the tag information is 600 bits, the length of a time slot is around 11623 μ s. The transmission rate of the reader is 26.5 Kb/s. The reader needs 37.74 μ s to send one bit. Thus, it takes nearly 3925 μ s (including the waiting time) to send a message of length 96 bits. For all the implemented protocols, the total number of parity check values m is assumed to be known a priori in order to derive a realistic baseline performance comparison. In the RIC protocol, the frame size will change based on the number of parity check values to be collected. The first round targets collecting m parity check values, and its frame has m slots. Each consequent round targets a smaller value of m , and consequently its frame size is reduced. All results are obtained by averaging over 100 simulation runs in order to obtain an accurate value of the statistics.

6.2. COLLECTION TIME COMPARISON UNDER DIFFERENT NUMBERS OF TAGS

The performance of the RIC protocol was evaluated under different group sizes n . The group sizes vary from 1000 to 5000. According to the requirement of group integrity of RFID tags, each tag in a group stores the same amount of parity check information j , which is set to 3 and 4 [8]. The value of P_{FP} is set to 1×10^{-4} to achieve the accuracy of 99.99%. Table II presents the results of the simulated protocols.

The collection times of all the protocols increase as the number of tags n increases. This is intuitive, because the tags need to transmit more information. The RIC protocol obviously outperforms other protocols, followed by the MIC protocol, the BC protocol and, finally, the EA protocol. For example, when the number of tags in a group is 3000 and the number of parity check values is 1800, the collection time of the EA protocol is 138.31s, which is around 24 times the lower bound. The BC protocol follows with 47.32s, and the MIC protocol further minimises the time to 33.63s. The RIC protocol considerably reduces the time of the MIC protocol by around 30% to 23.56s. Its collection time is about four times the lower bound.

Examining the second columns of $n = 3000$, where the number of parity check values is increased to 2400, shows that the EA protocol needs 183s. The BC protocol reduces the collection time by nearly 62% to 68.83s. The MIC protocol requires 45.32s, which is around six times the lower bound. The RIC protocol also considerably reduces the collection time to 32.06s. Again, the collection time of the RIC protocol is about four times the lower bound, and this conclusion can also be drawn from all columns of Table II.

Table II. Collection Time Comparison (in Seconds) For Different Group Sizes and Information lengths

n	1000		2000		3000		5000	
m	600	800	1200	1600	1800	2400	3000	4000
j	3	4	3	4	3	4	3	4
x	146		151		156		161	

Total tag information ($j \times x$) in bits	438	584	453	604	468	624	483	644
EA	30.31	40.1	76.86	101.6	138.31	183	278.11	367.84
BC	15.36	22.34	31.13	45.28	47.32	68.83	80	116.23
MIC	10.58	14.3	21.78	29.4	33.63	45.32	57.57	77.71
RIC	6.71	8.93	14.68	20.1	23.56	32.06	41.94	57.75
Lower bound	1.83	2.45	3.78	5.04	5.84	7.79	10.02	13.36

6.3. COLLECTION TIME COMPARISON UNDER DIFFERENT SETTINGS OF THE FALSE POSITIVE RATE

The relationship between the collection time of the RIC protocol and P_{FP} was also investigated. The EA, BC and MIC protocols do not have false positives. The focus here is on the RIC protocol. Table III compares the collection times required by the RIC under different values of P_{FP} ranging from 1×10^{-2} to 1×10^{-6} , where the group size is set to $n = 2000$. Generally, the collection time of the RIC protocol keeps increasing with the decrease in P_{FP} . For example, when P_{FP} is set to 1×10^{-2} in order to achieve accuracy of 99%, the collection time of the RIC reaches 10.19 s when $m=1200$. When P_{FP} is reduced to 1×10^{-3} in order to achieve accuracy of 99.9%, the time of the RIC protocol increases to 12.48 s. This behaviour is due to the fact that the relation between P_{FP} and b is inversely proportional, as presented in (7). That is, a low amount of P_{FP} causes a larger b . Consequently, a larger b in the RIC protocol increases the communication overhead and thus, the time needed for transmitting the BF vector between the reader and the tags. The collection time of the RIC protocol reaches a higher value equal to 19.45 s when P_{FP} is set to the lowest tested value, which is 1×10^{-6} , in order to achieve accuracy of 99.9999%. However, the RIC protocol is still faster than the collection times of the other simulated protocols as shown in the column of $n=2000$ with $m = 1200$ of Table II and the same conclusion can also be drawn when m is set to 1600.

Table III. Collection Time Comparison (in Seconds) Under Different Values of P_{FP} , Where $n = 2000$

P_{FP}	1×10^{-2}	1×10^{-3}	1×10^{-4}	1×10^{-5}	1×10^{-6}
Collection time when $m = 1200$	10.19	12.48	14.68	17	19.45
Collection time when $m = 1600$	14	17.01	20.1	22.83	26.26

7. CONCLUSIONS

This study aimed to efficiently collect redundant information from a group of tags in RFID systems. Existing solutions to general and different problems are not efficient since they do not distinguish the redundancy among tags, which results in long collection times. To achieve time efficiency, a new protocol called the RIC protocol is proposed. In this protocol, a BF is utilised as a tool to carry aggregated information in order to efficiently filter redundant tag information and thus save collection time. Extensive simulations were conducted to evaluate the performance of the proposed protocol. The results demonstrate that the proposed protocol outperforms other possible solutions by saving nearly 37%–84% of the collection time, which is around four times the lower bound. This makes the RIC protocol a promising candidate for collecting redundant tag

information in the group integrity of tags in RFID systems and other similar ones. For future work, the impact of channel error between the reader and the tags can be investigated and considered in the design of the RIC protocol.

REFERENCES

- [1] Nambiar, A. N. "RFID Technology: a Review of its Applications." *Proceeding of the World Congress on Engineering and Computer Science*, San Francisco, USA, 20–22 October 2009, pp. 1253–1259.
- [2] Sato, Y., Mitsugi, J., Nakamura, O., Murai, J. "Group Coding of RF Tags to Verify the Integrity of Group of Objects." *IEEE International Conference on RFID*, Orlando, USA, 12–14 April 2011, IEEE, pp. 200–206.
- [3] Sato, Y., Mitsugi, J., Nakamura, O., Murai, J. "Theory and Performance Evaluation of Group Coding of RFID Tags." *IEEE Transaction on Automation Science and Engineering*, 2012, 9, (3): pp. 458–466.
- [4] Sato, Y., Igarashi, Y., Mitsugi, J., Nakamura, O., Murai, J. "Identification of Missing Objects with Group Coding of RF Tags." *IEEE International Conference on RFID*, Orlando, USA, 3–5 April 2012, IEEE, pp. 95–101.
- [5] Su, Y., Lin, J., Tonguz, O. K. "Grouping of RFID Tags via Strongly Selective Families." *IEEE Communication Letters*, 2013, 17, (2): pp. 1–4.
- [6] Su, Y., Tonguz, O. K. "Using the Chinese Remainder Theorem for the Grouping of RFID Tags." *IEEE Transaction on Communications*, 2013, 61 (11): pp. 4741–4753.
- [7] Su, Y. S. "Extended Grouping of RFID Tags Based on Resolvable Transversal Designs." *IEEE Signal Processing Letters*, 2014, 21, (4): pp. 488–492.
- [8] Alkanhel, R., Ambroze, M. "Extended Grouping of RFID Tags Based on Progressive Edge-Growth Methods." *5th International Conference on Electronic Devices, Systems and Applications (ICEDSA)*. Ras Al Khaimah, UAE, 6–8 December 2016, IEEE, pp. 1–5.
- [9] Alkanhel, R., Ambroze, M. "Decoding Complexity Analyses of PEG-Based Methods with the Improved Hybrid Iterative/Gaussian Elimination Decoding Algorithm." *International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, Ras Al Khaimah, UAE, 21–23 November 2017, IEEE, pp. 239–243.
- [10] Chen, S., Zhang, M., Xiao, B. "Efficient Information Collection Protocols for Sensor-Augmented RFID Networks." *IEEE International Conference on Computer Communications*, Shanghai, China, 10–15 April 2011, IEEE, pp. 3101–3109.
- [11] Qiao, Y., Chen, S., Li, T., Chen, S. "Energy-Efficient Polling Protocols in RFID Systems." *International Symposium on Mobile Ad Hoc Networking and Computing*, Paris, France, 17–19 May 2011, ACM, pp. 1–9.
- [12] Qiao, Y., Chen, S., Li, T., Chen, S. "Tag-Ordering Polling Protocols in RFID Systems." *IEEE/ACM Transactions on Networking*, 2016, 24, (3): pp. 1548–1561.
- [13] Yue, H., Zhang, C., Pan, M., Fang, Y., Chen, S. "A Time-Efficient Information Collection Protocol for Large-Scale RFID Systems." *IEEE International Conference on Computer Communications*, Orlando, USA, 25–30 March 2012, IEEE, pp. 2158–2166.
- [14] Yue, H., Member, S., Zhang, C., Pan, M. "Unknown-Target Information Collection in Sensor-Enabled RFID System." *IEEE/ACM Transaction on Networking*, 2014, 22, (4): pp. 1164–1175.
- [15] Lee, S. R., Joo, S. D., Lee, C. W. "An Enhanced Dynamic Framed Slotted ALOHA Algorithm for RFID Tag Identification." *2nd International Conference on Mobile and Ubiquitous Systems Networking and Services*, San Diego, CA, USA, 17–21 July 2015, IEEE, pp. 166–172.
- [16] Cha, J. R., Kim, J. H. "Dynamic Framed Slotted ALOHA Algorithms Using Fast Tag Estimation Method for RFID System." *IEEE International Consumer Communications & Networking Conference*, Las Vegas, USA, 8–10 January 2006, IEEE, pp. 768–772.
- [17] Pan, L., Wu, H. "Smart Trend-Traversal: a Low Delay and Energy Tag Arbitration Protocol for Large RFID Systems." *IEEE International Conference on Computer Communications*, Rio de Janeiro, Brazil, 19–25 April 2009, IEEE, pp. 2571–2575.
- [18] Qian, C., Liu, Y., Ngan, R. H., Ni, L. M. "ASAP: Scalable Collision Arbitration for Large RFID Systems." *IEEE Transactions on Parallel and Distributed Systems*, 2013, 24, (7): pp. 1277–1288.
- [19] Wang, X., Ji, Y., Zhao, B. "An Approximate Duplicate Elimination in RFID Data Streams Based on D-Left Time Bloom Filter." *International Conference on Web Technologies and Applications*, Changsha, China, 23–25 September 2014, Springer, pp. 413–424.
- [20] Kamaludin, H., Mahdin, H., Abawajy, J. H. "Filtering Redundant Data from RFID Data Streams." *Journal of Sensors*, 2016: pp. 1–8.
- [21] EPCglobal. "Radio Frequency Identity Protocols Generation-2 UHF RFID Protocol for Communications at 860 mhz-960mhz." 2015, version 2.0.1.
- [22] Finkenzeller, K. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication*. Munich, Germany: Wiley, 2010.
- [23] Bloom, B. "Space/Time Trade-Offs in Hash Coding with Allowable Errors." *Communications of the ACM*, 1970, 13, (7): pp. 422–426.

- [24] Broder, A., Mitzenmacher, M. "Network Applications of Bloom Filter: a Survey " Internet Mathematics, 2004, 1, (4): pp. 485–509.
- [25] Geravand, S., Ahmadi, M. "Bloom Filter Applications in Network Security: A State-of-the-Art Survey." Computer Networks, 2013, 57, (18): pp. 4047–4064.