Computer Communications 88 (2016) 73–83

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Efficient tag reading protocol for large-scale RFID systems with pre-reading

Shuen-Chih Tsai*, Yu-Min Hu, Chen-Hsun Chai, Jung-Shian Li

Department of Electrical Engineering, Institute of Computer and Communication Engineering, National Cheng Kung University, University Road, Tainan 701, Taiwan, ROC

ARTICLE INFO

Article history:
Received 3 September 2015
Revised 19 March 2016
Accepted 23 April 2016
Available online 25 April 2016

Keywords:
RFID tag collection
STT
Blocking protocol
Missing-tag problem

ABSTRACT

In large-scale RFID systems, collecting all of the tag IDs is a time-consuming process. A protocol designated as Smart Trend-Traversal (STT) has been proposed to reduce collisions during the tag collection process and to dynamically construct the query strings used to interrogate the tags. In general, if the tag ID information is known to the reader from a previous tag collection round, the efficiency of the current round can be significantly improved. Various protocols have been proposed for scanning a known tag set based on the use of a hash function. Accordingly, the present study proposes an Enhanced STT scheme based on a blocking protocol and a Distributed Record Tag-Check (DRTC) mechanism. Compared to the conventional STT scheme, the proposed protocol adaptively adjusts the length of the query string depending on the response received to the previous query. Moreover, in the DRTC mechanism, the tags determine their transmission slot frame directly without the assistance of the reader, and thus the overall overhead of the tag-collection process is reduced. The simulation results show that the Enhanced STT scheme reduces the total number of queries required to collect the entire tag set compared to the conventional STT method. Moreover, the proposed DRTC mechanism yields an effective reduction in the total number of frame slots compared to existing protocols such as TPP/CSTR and ECRB.

1. Introduction

Radio frequency identification (RFID) technology [1] has many important applications nowadays , including item identification, automatic inventory and asset management, payment, and so on [9,24,37]. RFID systems have many advantageous features, including a relatively low deployment cost, an ability to operate in harsh environments, the absence of batteries or any form of external energy source, and so forth. However, in implementing RFID systems, two problems arise, namely the RFID tag-collection problem [16] and the RFID missing-tag event problem [21,29]. Various protocols have been proposed for improving the efficiency of the tag ID collection process. Moreover, various methods have been developed for solving the missing-tag problem, i.e., detecting the missing tags among a set of known tags. However, in previous study, they usually either process the tag identification collection or detect the missing tags independently. If the process of tag identification finishes, the process of missing-tag detection will start separately. In general, the system spends some time on switching these two processes. Our proposed protocol will switch the processes smoothly. We want to solve these two problems together and let our proposed scheme more efficient.

The present study proposes an Enhanced Smart Trend-Traversal (Enhanced-STT) scheme based on the STT protocol [18] and together with a blocking mechanism [35] and a Distributed Record Tag-Check (DRTC) mechanism. The conventional STT protocol uses a depth-first-search method to construct the query traverse path. Moreover, when collisions occur, the number of bits appended to the query string used to interrogate the tags is dynamically adjusted in accordance with the tag ID distribution or the tag population. However, in the Enhanced STT scheme, the query length is adjusted dynamically in accordance with the response received from the previous query in order to accelerate the tree traversal process. In addition, in the DRTC mechanism, the tags determine their transmission frame-slots directly from an inspection of the query length. In other words, the reader is not required to transmit slot information to them, and thus the tag collection overhead is reduced. Compared to traditional binary tree-based protocols [34], DRTC has a lower system requirement (i.e., a fewer number of counters per tag), but retains an excellent performance.

The simulation results confirm that the Enhanced STT protocol reduces the total number of queries required to collect the entire tag set compared to STT. Moreover, the Enhanced STT scheme

---

* Corresponding author. Tel.: +886 6 2757575x62400.
E-mail address: pandaorz@gmail.com (S.-C. Tsai).

http://dx.doi.org/10.1016/j.comcom.2016.04.019
0140-3664/

reduces the number of collisions in the tag-collection process, and therefore reduces both the total tag identification time and the energy consumption of the active tags. The efficiency of the Enhanced STT scheme is further evaluated in terms of both the System Efficiency (*SE*) and the Time System Efficiency (*Time_SE*). It is shown that the *Time_SE* of Enhanced STT is superior to that of STT and other Query Tree-based (QT-based) protocols. In addition, it is shown that the DRTC mechanism provides an efficient approach for checking the IDs of staying tags and reduces the total number of frame slots required to collect all of the tag IDs compared to existing protocols such as Three-Phase Protocol with Collision Sensitive Tag Removal (TPP/CSTR) [21] and Enhanced Couple-Resolution Blocking (ECRB) [35].

The remainder of this paper is organized as follows. Section 2 reviews the related work in the RFID tag identification field. Section 3 introduces the Enhanced STT scheme and DRTC mechanisms proposed in this study. Section 4 presents and discusses the simulation results Finally, Section 5 provides some brief concluding remarks and indicates the intended direction of future research.

2. Related works

Research in the RFID field focuses predominantly on two main problems, namely the RFID tag anti-collision problem and the RFID missing-tag event problem. The former problem involves improving the efficiency of the tag collection problem by avoiding collisions [16]. The later problem involves scanning a large group of known tags and identifying any tags which are missing in the current scanning round [4]. The present study proposes an approach for reducing the tag identification time by extending the conventional STT algorithm. STT is a QT-based anti-collision protocol. Thus, the following sub-section introduces previous work relating to QT-based protocols. Section 2.2 then reviews existing proposals for dealing with the missing tag event in the second round (and on) of the tag-collection process.

2.1. RFID tag anti-collision protocols

Existing RFID tag anti-collision protocols generally adopt either a tree-based approach [11,15,23,25,26,34–36] or an Aloha-based approach [3,7,8,14,19,20,22,30–32]. Because of scalability in real life [5], our study adopts tree-based approaches. Among protocols adopting the former method, the Query Tree (QT) algorithm [2,6], and Binary Tree (BT) algorithm [34] are among the most common. In the QT algorithm, the RFID reader uses the tag ID prefix to perform the tag-collection process. By contrast, the BT algorithm identifies the tags by using random binary numbers to partition the RFID tags into small groups. Meanwhile, in Aloha-based protocols, the reader first estimates the number of tags within its communication range and then assigns these tags an appropriate number of frame slots such that they can return their IDs without contention [28]. We adopt the mechanism without estimating the number of tags, because it is more easy to be deployed in real life.

Irrespective of the anti-collision protocol employed, it is necessary to deal with two different types of tags, namely staying tags and arriving tags. Staying tags are tags identified by the reader in a previous tag-collection round which are also present in the current round. By contrast, arriving tags are tags which appear in the reader's communication range for the first time in the present round. In other words, they are unknown to the reader prior to the current round. As described above, in attempting to improve the efficiency of the tag-collection process, the present study extends the conventional QT-based STT protocol. Thus, the following discussions briefly review the original QT algorithm and its variants.

2.1.1. QT protocol and variants

In the QT protocol, the tag set is split using the tag ID prefix in order to construct a query tree. Having constructed this tree, the reader then broadcasts tag ID queries in a breadth-first manner. The reader maintains a queue (*Q*) of query strings and initializes this queue with two 1-bit strings, i.e., 0 and 1, at the beginning of each tag-collection round. The reader then de-queues a query string from *Q* in order to interrogate the tags within its communication range. Any tags having a prefix which matches the query transmit their IDs to the reader. If the tag IDs arrives at the reader in different time slots, they can be successfully recognized. However, if multiple tags respond simultaneously, the reader detects a collision event and splits the original query $b_1b_2\dots b_{lc}$ into two sub-tree queries where $b_i \in \{0, 1\}$, i.e., $b_1b_2\dots b_{lc}0$ and $b_1b_2\dots b_{lc}1$, which it then enqueues to the end of its queue. In the event that the reader detects just one tag response or no tag response, referred to as a single-tag-response query and an idle query, respectively, it concludes that query $b_1b_2\dots b_{lc}$ is the end of the current sub-tree, and labels the query string "end" accordingly. The reader then dequeues the next query string from *Q* and repeats the interrogation process. The process is repeated iteratively in this way until all of the query strings in *Q* are labelled as "end".

In the QT-based Adaptive Query Splitting (AQS) protocol [13], the single-tag-response and idle queries obtained in the previous tag-collection round are stored by the reader in a candidate queue (*CQ*). At beginning of each tag-collection round, the queries in *CQ* are copied to *Q* and are then sequentially dequeued in order to identify the staying tags and arriving tags, respectively. The Couple-Resolution Blocking (CRB) protocol [35] extends AQS by introducing a blocking mechanism. In the proposed approach, the reader "mutes" arriving tags as it is identifying the staying tags in order to prevent collisions between them. CRB distinguishes between the two types of tags by allowing the reader and all the tags to store both the last reader ID and the frame number. CRB divides the tag-collision process into two phases, namely a staying tag identification phase (Phase 1) and an arriving tag identification phase (Phase 2). In Phase 1, the reader dequeues two queries from *Q* and transmits a concatenated query including both queries. Since the arriving tags are muted in Phase 1, the reader interprets the occurrence of a collision event as meaning that the two tags are still present (i.e., they are staying tags). Importantly, this "2-collision" concept yields a significant reduction in the time required to identify the staying tags compared to the original AQS protocol. Enhanced CRB combines two similar prefixes into one query. Thus, sending this query can further reduce the bits sent by reader.

2.1.2. Smart trend-traversal protocol

Various protocols based on QT have been proposed in which the reader broadcasts queries in a depth-first rather than breadth-first manner [10,18,33]. In such protocols, when a query $b_1b_2\dots b_{lc}$ results in a collision, the reader infers that the current query is at a higher bit-level in the query tree, and thus the next query is set to $b_1b_2\dots b_{lc}0$ such that the reader traverses down to the child node in the query tree. In the event of a single-tag-response, the next query is set as $(b_1b_2\dots b_{lc})+1$ and traverses horizontally to the node located immediately to the right of the child node in the tree. Finally, if an idle query occurs, the next query is set to $b_1b_2\dots b_{lc-1}1$ if $b_{lc}=0$ (i.e., b_{lc} is a left child node), or $b_1b_2\dots b_{lc-1}$ if $b_{lc}=1$ (i.e., b_{lc} is a right child node), such that the query traverses in the right upper direction and converge to the next prefix node. Fig. 1 presents an illustrative example of the STT protocol. There are 5 tags in the query tree. Each node represents the status of a query, including collision, single-tag-response and idle. Note that the numbers on the graph show the sequence step in the STT process.