exeley INTERNATIONAL JOURNAL ON SMART SENSING AND INTELLIGENT SYSTEMS

Article | DOI: 10.21307/ijssis-2021-009 Issue 1 | Vol. 14 (2021)

TDMA-clustering-based approach to avoid the reader-to-tag collision problem during the stocktaking process

Sara El Ouadaa*, Slimane Bah and Abdelaziz Berrado

Équipe AMIPS, École Mohammedia d'ingénieurs, University, Mohammed V, UM5, Rabat, Morocco.

*E-mails: saraelouadaa@research.emi.ac.ma; sara.elouadaa@gmail.com

This paper was edited by Subhas Chandra Mukhopadhyay.

Received for publication November 5, 2020.

Abstract

The reader-to-tag collision problem occurs when multiple readers try to access the same tag simultaneously. The traditional collision avoidance techniques such as RTS (request to send) and CTS (clear to send) are not applicable because a reader may communicate with multiple tags simultaneously. In this paper, we introduce a collaborative communication protocol to avoid reader-to-tag collisions using TDMA and clustering approaches. The protocol targets the RFID-WSN static systems arranged in a square grid topology, which we can find in different RFID applications such as warehouse stocktaking, parking cars, agricultural fields, and libraries. In such simple topologies, the other proposed reader collision solutions for general use of RFID systems are not efficient since they cannot avoid all possible collisions, and worse of that, some of them are not even detectable, which is intolerable for stocktaking applications. Moreover, they are complicated and heavy in resources, while read throughput is limited. Our protocol presents a simple solution for simple RFID systems with better performances. To validate the proposed protocol, we presented a model using the Process Meta Language (Promela), which is executed under the simple Promela interpreter (SPIN) model checker to verify the protocol properties as deadlocks and livelocks. Also as a proof of concept, we have done a first-step performance analysis using the java runtime.

Keywords

RFID, Reader Collision, Communication Protocol, WSN, Clustering, TDMA.

Static RFID systems have a wide range of uses in warehouses, libraries, parking cars, and some areas of the agriculture industry (Ali et al., 2017; Krebs and Liard, 2001; Mekala et al., 2017). In some cases, they can be combined to wireless sensor networks to complete their sensing and computation capabilities, the work in Nagpurkar and Jaiswal (2015) presents the four possible combination forms, which are integrated tags with sensor, integrated tags with wireless sensor nodes, integrated readers with wireless sensor nodes, and the mixed architecture.

An RFID system consists of RFID tags and networked RFID readers. The tags may be active equipped with a battery, or passive with no explicit power supply (Finkenzeller, 2010). To query the stored information from a passive tag, the reader transmits a high-power continuous wave to energize the tag. The tag receives the energy and transmits the stored information by backscattering communication with the reader. Passive tags are mainly used because they are economically affordable. The readers have two different ranges, namely the interrogation range and the interference range (Engels and Sarma, 2002). The interrogation range can be defined by the maximum distance from which a reader can read surrounding tags and the interference range can be defined by

the maximum distance from which the emitted signal of a reader interferes with the one of another reader. Since the signal from a passive tag to the reader is a reflected signal, the reader interrogation zone is very limited. Therefore, in some applications, several readers' aggregation in the RFID system comes to be necessary to cover a target area.

In such dense environments, the RFID readers must be arranged such that all tags, regardless of where they are within the target area, can communicate with at least one reader, and therefore the interrogation and interference range of adjacent readers intersect, causing collisions in the system. These collisions imply failure in recognizing tags and inevitably decrease the performance of the system. Thus, one of the main goals of research works in the field of RFID is to find a solution for the collision problem. There exist two types of reader collisions, which are discussed below:

1. Reader-to-tag collision: occurs when two or more readers interrogate the same tag simultaneously as their read ranges intersect, as shown in Figure 1(a).
2. Reader-to-reader collision: occurs when the signal emitted by one reader interferes with the reception system of the others, as shown in Figure 1(b).

Standard multiple access schemes such as Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA), Currier Sense Multiple Access (CSMA), and Code Division Multiple Access (CDMA) cannot be directly applied to RFID systems (Joshi and Kim, 2008) because of the following problems as mentioned below:

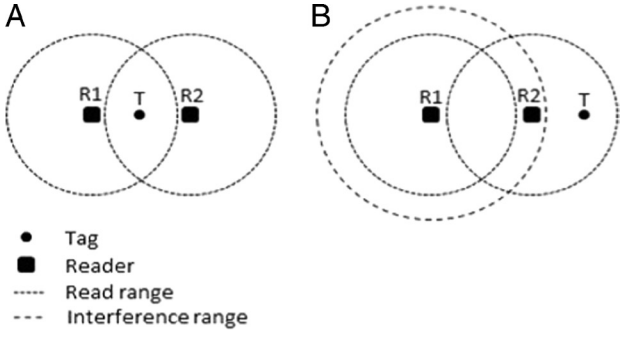

Figure 1: (a) The readers-to-tag collision: the collision happens when R1 and R2 interrogate T simultaneously. (b) The Reader-to-reader collision: R1 interference range affects R2 reading range and hence R2 cannot read T.

In FDMA, the interfering readers use different frequencies to communicate with the tags. As RFID tags cannot choose a particular frequency, they cannot choose a particular reader to establish a communication link.

In TDMA, the interfering readers are allocated different time slots, to avoid simultaneous transmissions. However, the system needs a tight time synchronization to determine the start time slot of all readers.

In the CSMA scheme, before transmitting, the readers listen to the channel to detect whether it is busy or idle to avoid collisions. However, carrier sensing is not very effective especially in dense environments as it cannot avoid collisions when more than one reader wants to transmit at the same time, also constantly sending beacons to avoid the collisions is poor energy efficiency (Mbacke et al., 2018). Furthermore, when using different channels, reader-to-tag collisions will occur, because the readers cannot detect the received signal from its neighbors.

On the basis of the pseudo-random codes, the CDMA scheme requires extra circuitry at the tags, which is not cost-effective for low-cost practical RFID tags. It would also complicate the deployment phase by assigning the codes to all tags.

In this paper, we particularly address the reader-to-tag collision problem in a static RFID-WSN system dedicated to the stocktaking process. We propose a decentralized mechanism that uses the TDMA scheme and the clustering approach to permit communications between interfering readers. In so doing, we developed a communication protocol to be implemented in the network's cluster heads to provide the readers' synchronization and coordinate their communications. To avoid collisions between adjacent clusters, we added collaboration functionality to the designed protocol that allows scan synchronization between adjacent clusters.

The rest of the paper is organized as follows. In the second section, we present prior works; we describe then the proposed protocol in the third section, while in the fourth section, we show the protocol validation using the SPIN model checker. Finally, the last section provides a short overview of our proposal and an outlook for future work to progress toward a full solution.

Related work

In the work of Joshi and Kim (2008), a classification of the reader collision solutions was presented; it describes three different categories: control-based,

coverage-based, and schedule-based solutions. The control-based solutions mitigate the problem of collisions between readers, by transmitting notification control packets such as beacon signals. After receiving a beacon signal, the interfering readers interrupt their ongoing communication and wait for the next cycle. However, this kind of solution cannot avoid the collision as beacons of competitor readers may collide.

The coverage-based solutions consist of dynamically adapting the read ranges to reduce the overlapped areas between adjacent readers. It relies on two different approaches: the clustering approach, where a cluster head is elected to adjust the read ranges, and an adapted transmission approach, which usually needs a central node to calculate the distance between each pair of readers and adjust their reading ranges. However, reducing the read range in dense tag environments increases the likelihood of uncovered tags. While the schedule-based solutions consist of allocating the available system resources such as the frequencies and time among the readers to prevent them from colliding. Colorwave (Waldrop et al., 2003), the Neighbor Friendly Reader Anti-Collision algorithm (NFRA) (Eom et al., 2009), the Distance-based Reader Collision Avoidance algorithm (DRCA) (Golsorkhtabaramiri and Issazadehkojidi, 2017), and the Fair Reader Collision Avoidance Algorithm (FRCA 1&2) (Rezaie and Golsorkhtabaramiri, 2018) are examples of the scheduling-based approach.

More broadly, authors in Mbacke et al. (2018) have divided the reader collision solutions into two main categories: namely distributed and centralized mechanisms. In distributed mechanism, the readers communicate with each other; usually wirelessly, to share the resources and maintain the synchronization. However, this type of mechanism requires the system to establish and maintain information over the network, which is time and energy-consuming. Colorwave (Waldrop et al., 2003), the Distributed Tag Access with Collision-Avoidance algorithm (DiCa) (Hwang et al., 2006), Pulse (Birari and Iyer, 2005), the Multi-Channel MAC algorithm (MCMAC) (Dai et al., 2007), the algorithm proposed in the paper (Golsorkhtabaramiri et al., 2015), the Distributed Multi-Channel Collision Avoidance algorithm (DIMCA) (Safa et al., 2015) and the Efficient Multichannel Reader Collision Avoidance algorithm (EMRCA) (Jiang et al., 2016) are examples of distributed protocols.

Whereas in a centralized mechanism, a central server is in charge of reader coordination, it may communicate to the readers either in a wired or wireless way to synchronize them and share the available resources among them. Pulse Protocol (Birari and Iyer, 2005) is a distributed protocol that comes under the control-based category, based on a beaconing mechanism, while reading a tag the reader broadcasts periodic beacon messages on a separate control channel. Before a competitor reader can scan a tag, it first senses the control channel for a beacon. If it does not receive any beacon for a specified amount of time, it transmits a beacon and starts the scan. Pulse protocol spends so much energy for information transmission over the control channel; its periodic transmission of beacons increases the overhead. Furthermore, it cannot solve the hidden terminal problem (Joshi and Kim, 2008). Colorwave (Waldrop et al., 2003) is a distributed algorithm and comes under the scheduling-based category, in this algorithm, each timeslot is allocated with a different color, where readers randomly select a timeslot among a dynamic range of available colors. The range of colors varies according to the network situation; each reader monitors its number of successful interrogations and accordingly modifies its local value of maximum available colors. As such, if more than one reader selects the same color, a collision occurs. In this case, involved readers randomly choose a new color and reserve it for the following interrogation round by sending a kick message to the neighbors. All readers on the corresponding color have to switch to a different timeslot for the following round. The readers maintain synchronization among each other by continuously tracking the current time slot. Hence, the system overhead is significantly impacted.

In centralized NFRA protocol (Eom et al., 2009), the readers randomly choose a time slot from the available ones broadcasted in the first place by the server. The readers elected for communication are the ones whose time slot corresponds to the single time slot broadcasted by the server at a second place. Before tag scanning, the readers send a beacon message to detect collisions between elected readers. If no collision occurs the reader sends a message to prevent the competitor readers from election until the next round. NFRA has a major drawback; it is unfair on sharing the available time slots among the readers since in each round only readers having fewer neighbors are selected (Rezaie and Golsorkhtabaramiri, 2018). FRCA (Rezaie and Golsorkhtabaramiri, 2018) came to address the lacuna observed in NFRA, it also brings the FDMA mechanism. The authors proposed two versions of their algorithm. In FRCA1, the same scheme of NFRA is followed having a central server broadcasting commands and the readers randomly choosing timeslots and channels and sending beacons. In case of a beacon collision, instead of both readers getting

disabled as in NFRA, they compare their number of successful transmissions, as the reader with the lowest success rate will start tag interrogation and the other reader will wait for the next round. This approach adds fairness between readers for getting access to the medium. However, the algorithm raises the reader-to-tag collisions, as readers operating in different frequencies, can communicate at the same time. To address this kind of collision, the authors also proposed FRCA2 where involved readers will not only compete upon their reading success rate but also according to the distance between them. If the distance between the two readers is less than two times the reading range, the loser reader having a larger success rate will choose another channel and wait for the next round. While it will get access to the medium on the next time slot and using a different channel if the distance between them is larger than two times the reading range.

DRCA (Golsorkhtabaramiri and Issazadehkojidi, 2017) protocol is yet another centralized protocol that aims to avoid the reader collisions based on the measured distance between readers. A polling server that broadcasts an AC packet at the starting point of each round arranges the read rounds. The AC packet determines the number of available time slots. The readers randomly choose a time slot and select one of the four suggested channels by ETSIEN 302 208, randomly. After choosing a time slot, readers decide according to the channel situation in the previous time slot. By listening to the channel the reader knows if it was busy or not, if the channel was free in the previous time slot, the reader will broadcast a beacon packet if no collision occurs the reader can start communication with the tag. Conversely, if the reader detects that the channel was busy in the previous slot, it will decide according to the distance between itself and the other reader. If the distance is more than two times the read range, the reader chooses another channel and competes in the next time slot, while if the distance is less than two times the read range, the reader also chooses another channel but competes in the next round. DiCa (Hwang et al., 2006) is a distributed collision avoidance algorithm. As Pulse protocol, it also has a data channel and a control channel. Each reader contends for the use of the data channel through the control channel. The winner reads the tags through the data channel, while the others wait until the channel is idle. The readers exchange the following packets for collision avoidance:

- BRD_WHO: Packet used for identifying whether a reader reading tags exists in the same network or not.
- BUSY: Used for indicating whether the reader is reading tags.
- BRD_END: Packet used for indicating that the channel is idle after the tags have been read.

DiCa considers the hidden and exposed terminal problems by adjusting the control channel range at twice the radius from the first reader. However, DiCa has some shortcomings. It causes a delay in the system by exchanging the contention messages. Also, it tries to solve the collision problem after it takes place, rather than acting preemptively. Thus, it cannot solve the collision problem completely. MCMAC (Dai et al., 2007) is yet another contention-based MAC protocol for RFID systems. Similar to the Pulse protocol, MCMAC reserves a control channel as a sub-band of the RFID spectrum for reader-to-reader communication. The readers can communicate simultaneously with the data channel and control channel.

MCMAC works similarly to the conventional LBT. MCMAC broadcasts a control message after it wins contention in a control channel and gains access to the data channel. The control message informs other neighboring readers within the interrogation zone that the particular channel is occupied for a certain time. At the reception of a control packet from a neighboring reader, the other readers will not use that channel for a certain period and try to gain access to another channel.

Even though this approach can mitigate the reader-to-reader problem, it cannot solve the reader-to-tag problem. Passive RFID tags are unable to differentiate between two data channels. Therefore, multiple data channels are not applicable in a passive tag environment. DiMCA (Safa et al., 2015) is another distributed protocol, it aims to avoid reader collisions based on a notification mechanism. For that, it uses two different control channels operating at different ranges. The first one covers the reading range of the reader and carries messages containing the ID of the reader, and the second channel covers the interference range and carries messages containing the reader's ID and its chosen channel. The two types of messages are kept in two different queues, the first one holds the IDs of neighbors susceptible to cause reader-to-tag collisions and the second one holds the IDs and chosen channel of neighbors that may cause reader-to-reader collisions.

A reader waits for a random time before interrogating tags. When its waiting time expires, the reader checks its queues and decides accordingly. If the first queue is empty, the reader chooses a different channel to operate on and broadcasts it to its neighbors beforehand, otherwise, it waits for an

END signal from its neighbors to operate at a different time.

Despite the improvement of both the throughput and efficiency of the RFID system, this solution relies on an overhead created by the exchanged messages, which can affect the delay. Furthermore, the authors did not address collisions among exchanged messages.

EMRCA (Jiang et al., 2016) protocol comes to improve the pulse protocol, it takes into account the multichannel aspect by identifying two types of collisions caused by the interrogation and interference range of readers. Readers sense first the common control channel used by all nodes to communicate. If the channel is idle during a given period, the reader begins the contending phase. Otherwise, depending on the neighbor on activity, it either starts a new listening session at the end of the current activity or, pursues the timer before contending. During contention, readers wait for a random delay time. If a reader receives a beacon during this time, it re-senses the control channel, otherwise, if the delay time runs out without any reception of the beacon, the reader moves into tag interrogation. It then gets access to the selected interrogation channel and periodically sends out a beacon to advertise on the common control channel. While this protocol improves the overall fairness and efficiency of Pulse, it still suffers from some shortcomings as readers' mobility and high density of readers' deployment.

To the best of our knowledge, only Golsorkh-tabaramiri et al. (2015) has addressed the reader collision problem in RFID–WSN integrated systems. It adapted Pulse protocol to the RFID enhanced wireless sensor network to consume less network bandwidth. A reader in communication with a tag periodically sends beacon packets on the common control channel. The beacon message contains the ID of the read tag. To avoid the same tag readings by multiple readers, each reader who receives the beacon message buffers a list of reading tags in each round. Before sending a beacon message, the reader starts by sensing the control channel. If it is busy, the reader pursues the sensing, if the channel is empty the reader waits for a time before sending the beacon message. However, this protocol generates a high overhead caused by the periodically broadcasted beacons.

To summarize, the previously proposed solutions for the general use of RFID technology are not best suited for a stocktaking use case. None of them has succeeded to register no collisions and no missed tags as depicted in Mbacke et al. (2018), while the stocktaking process is intolerant to such problems. Therefore, we came with a personalized solution for the stocktaking process to guaranty that all tags within the warehouse area will be read. It can also serve other applications having the same system topology. Our protocol comes under a decentralized mechanism in which distributed coordinators are in charge of arranging the operation of readers. As the coordinators are aware of the readers' positions, they will provide an efficient resource allocation capable of avoiding all possible collisions.

The proposed protocol

The proposed protocol was designed to avoid the reader-to-tag collisions in a warehouse equipped with an RFID network where each rack of a shelf is mounted with an RFID reader forming a square grid topology. Our system consists of four types of devices: WSN cluster heads, where each cluster head communicates to its directly connected neighbors, integrated readers with sensor nodes (IRs), ordinary RFID tags, and the base station (Nagpurkar and Jaiswal, 2015). The integrated readers are arranged in clusters where the cluster heads (CHs) dynamically allocate the available time slots among their cluster members, so when they receive the scan request from the base station, they will sequentially relay it to the corresponding readers.

In a square grid topology, the IRs are lined up in rows and in columns such that IRs within a column are a distance x apart and the IRs in the position I within adjacent columns are a distance x apart as shown in Figure 2. Let us assume that the potential collision may occur between readers that are at most distance $\sqrt{2}x$ apart (i.e. R1, R2, R3 and, R4 of each cluster in Figure 2), four time slots will be needed to avoid the collision (Engels, 2002). As the CHs are aware of the IRs locations, they will activate them in a way to avoid collisions. When R1 is activated, readers R2, R3, and R4 are disabled. The time slot ends when a reader has scanned all tags within range. This method guaranty to avoid all possible collisions and hence an efficient stocktaking.

The default order of the four time-slots allocation of each CH (Hereafter called scan sequence) will be S1, S2, S3, and S4. To avoid the collisions among adjacent clusters during a selective scan, this order may change. The involved CHs of a selective scan are responsible for readjusting their scan sequence order according to their need; this will be detailed in section "Cluster head behavior" below.

Protocol's exchanged messages

The proposed protocol allows CH collaboration through a set of exchanged messages which are

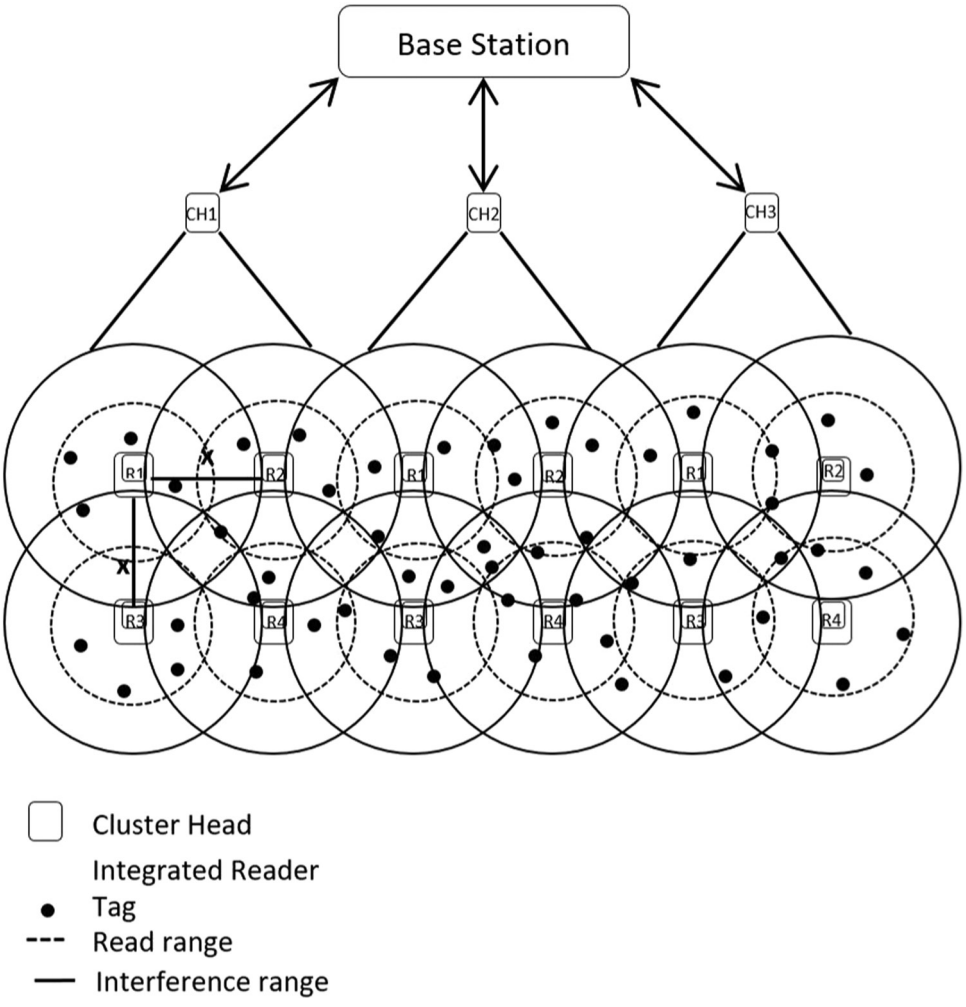

Figure 2: Deployed RFID architecture of the stocktaking use case.

presented in Table 1. We acknowledge almost all requests by a simple response "OK".

Cluster head behavior

In the general scan, where all the CHs receive the scan request at the same time, they follow the default scan sequence by first triggering the S1 members to start reading the tags. When the S1 members read all the tags within their read ranges, the CHs then trigger the S2 members, and so on until the end of the scan sequence. While in the selective scan, where the CHs receive the scan request at different times, each CH first checks the status of its adjacent neighbors to verify whether they have a scan in progress. If so, the CH will collaborate with the involved neighbors to start their common scans simultaneously. For that, the CH will redefine the necessary scan sequences and synchronize the scan among adjacent neighbors. While if no neighbor has a scan in progress, the CH keeps its default scan sequence and starts the scan. The adjacent CHs notify each other of the start and end of the scan. As so, all the CHs are permanently aware of their neighbors' status. The automaton in Figure 3 describes the general behavior of the proposed protocol.

In the selective scan, if the requested CH (e.g. CH2) has only one neighbor on scanning state (e.g. CH1), he will ask him for his remaining scans by sending him a remaining scan (remS) request. When the CH1 terminates the current scan (e.g. S1), he receives the request, replies to CH2 with a scan report (scanRep) message that contains the list of its remaining scans – S2, S3, S4 – and interrupt the scan until further notification from CH2. When CH2 receives the (scanRep) message it redefines its scan

Table 1. Set of the Protocol's messages.

Message	Semantic	Response
scnState	Inform the neighbors of start of scan	Ok
snsState	Inform the neighbors of end of scan	Ok
remScan	Ask the neighbor for his list of remaining scans	scanRep
continue_scan	Ask the neighbor to continue its regular scan	Ok
continue_scan_newSeq	Send the neighbor it's updated scan sequence and ask him to continue the next scan	Ok
start_double_scan	Ask the neighbor to start synchronized double scan	Ok
continue_double_scan	Ask the neighbor to continue next synchronized double scan	Ok
start_triple_scan_newSeq_sesMem	Send the neighbor it's updated scan sequence and the other neighbor's address, and ask him to start the synchronized triple scan	Ok
start_triple_scan_sesMem	Send the neighbor the other neighbor's address and ask him to start the synchronized triple scan	Ok
continue_triple_scan	Ask the neighbor to continue next synchronized triple scan	Ok
EOCS	Inform the neighbors of end of current scan	Ok

sequence to start with CH1's remaining scans – S2, S3, S4, S1 – and notifies him to start a synchronized double scan by sending him a (continue_double_scan) message. The synchronized scan allows the involved CHs to perform the common scans simultaneously. When the current scan is over, the CHs notify each other to start the next scan. Figure 4 describes the double scan coordination process. When CH2 has two neighbors in scanning state, he will send them both (remS) request and

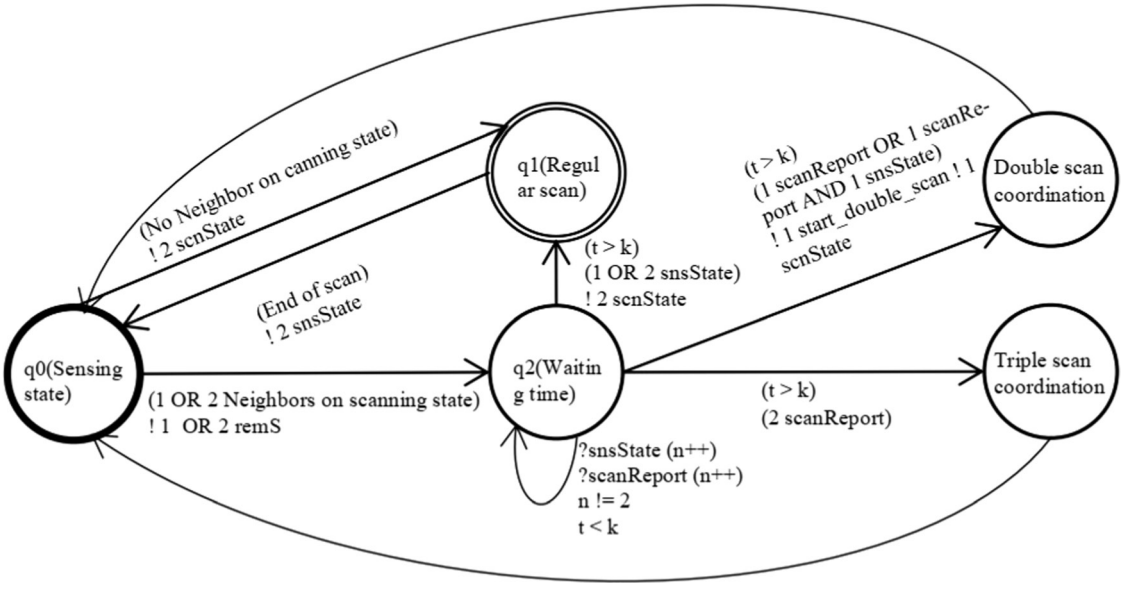

Figure 3: General automata for the proposed protocol.

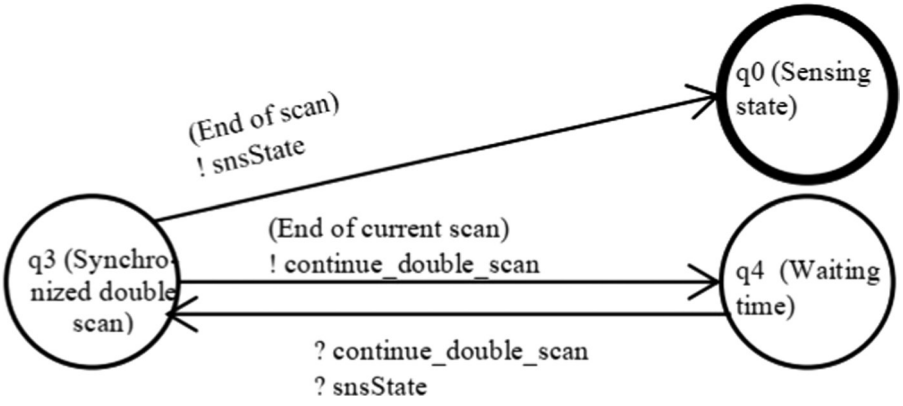

Figure 4: Double Scan Coordination.

waits for their (scanRep) messages, once he receives the lists of their remaining scans, he checks if they have some scans in common; four possibilities come up: (1) The remaining scans of both neighbors are all common. (2) The remaining scans of only one neighbor are all common. (3) The two neighbors have common and uncommon scans. (4) The two neighbors have no common scans. The flowchart of the proposed protocol is shown in Figure 5.

In the first case, where the remaining scans of both neighbors are all common (e.g. S2, S3, S4 for CH1 and S2, S3, S4 for CH3), CH2 redefines its scan sequence as to start with the common scans – S2, S3, S4, S1 – and notifies the neighbors to start the synchronized scan by sending them a (start_triple_scan_sesMem) message. The session member (sesMem) parameter introduces the distant members to each other – CH1 and CH3 –.

In the second case, where the remaining scans of only one neighbor are all common (e.g. S2, S3, S4 for CH1 and S3, S4 for CH3). CH2 redefines CH1 and its scan sequences – S3, S4, S2 and S3, S4, S2, S1, respectively, – he then notifies the neighbors to start the synchronized scan by sending CH3 a (start_triple_scan_sesMem) message, and CH1 a (start_triple_scan_newSeq_sesMem) message that contains its new scan sequence.

In the third case, where the two neighbors have common and uncommon scans (e.g. S3, S4 for CH1 and S4, S1 for CH3) – CH3, in this case, is supposed to be part of another synchronized scan – the CH2 will arrange the two neighbors to start with their uncommon scans and wait for them to run the common scan together. To do so he will redefine the neighbors' scan sequences to start with the uncommon scans and send them the new scan sequences in a (continue_scan_ newSeq) message, while he will redefine its sequence to start with the common scans – S4, S1, S2, S3 –. Once CH2 is notified by his neighbors of the end of their current scan (EOCS message), he will launch the synchronized scan by sending the neighbors a (start_triple_scan_sesMem) message.

In the fourth case, where the two neighbors have no common scans, the CH2 will also check if the two neighbors have the same number of remaining scans. If so (e.g. S4 for CH1 and S1 for CH3), he will ask the neighbors to continue their remaining scans by sending them a (continue_scan) request, while he will start his scan sequence after the two neighbors terminate theirs. If one neighbor has more remaining scans than the other (e.g. S3, S4 for CH1 and S1 for CH3), the CH2 will redefine its scan sequence to start with the extra scan of CH1 – S4, S1, S2, S3 – and ask the neighbors to continue their remaining scans. When the neighbor with fewer remaining scans – CH3 – terminates its sequence, CH2 will start a synchronized double scan with CH1. Figure 6 describes the triple scan coordination process.

Protocol validation

To validate the proposed protocol, we have done a model using the Process Meta Language (Promela), which is executed under the simple Promela interpreter (SPIN) model checker to verify the protocol properties such as deadlocks and livelocks, which is by default considered as the unintended end state of the system. We also presented a proof of concept of the proposed protocol, by going through a first step

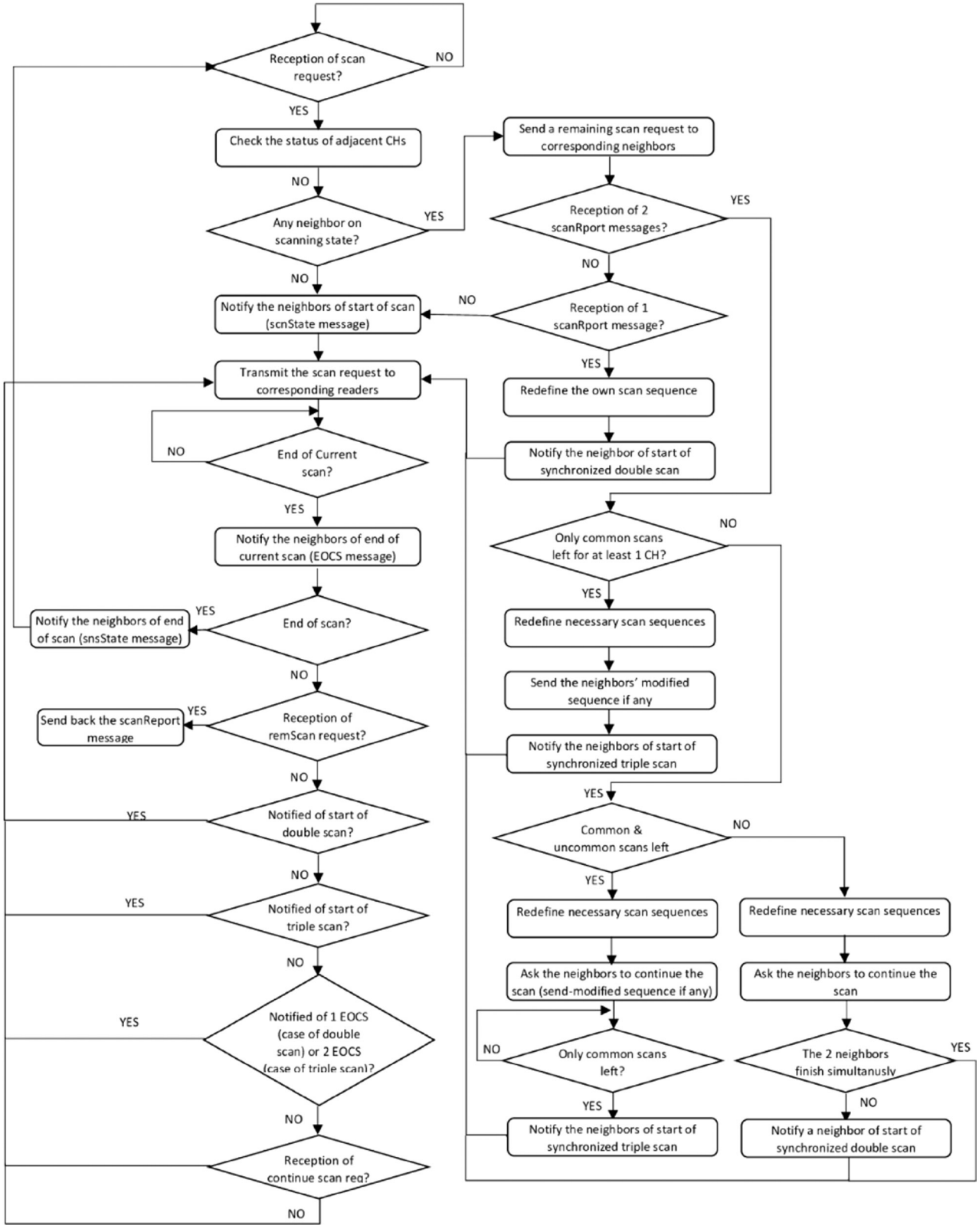

Figure 5: Flowchart of the proposed protocol.

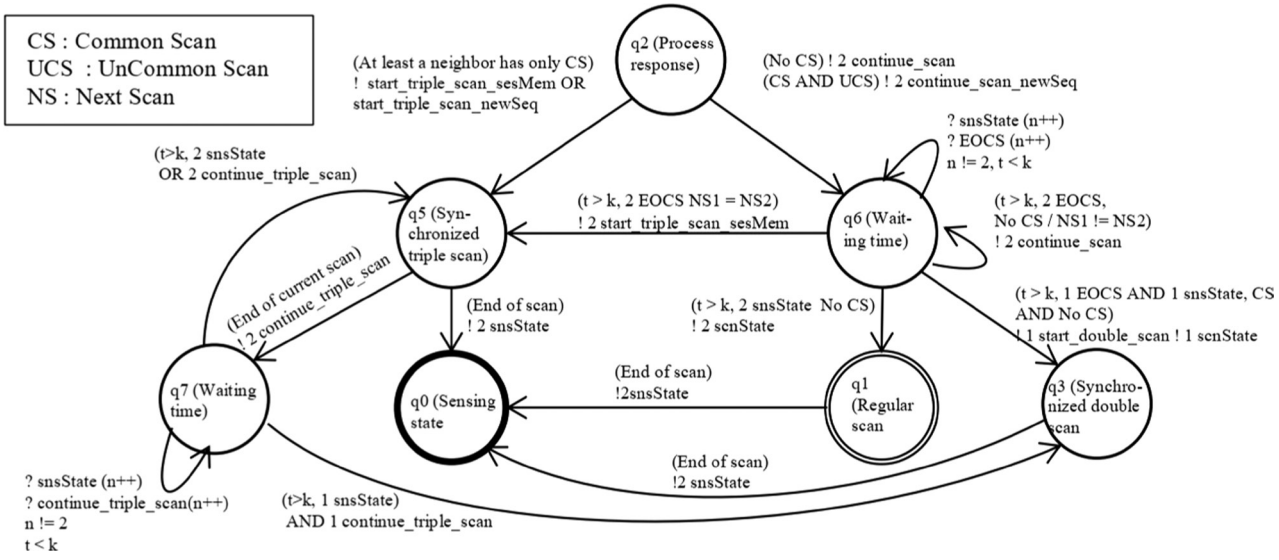

Figure 6: Triple scan coordination.

performance analysis using the java runtime, for that, we created a simple LAN network composed of two machines to measure the protocol performances such as response time of each packet and the system overhead. We have used UDP as a transport protocol, given the small amount of data to transmit, the three-way connection TCP protocol would be too heavy. The first machine is a Windows 10, Intel Core i5-8265U @ 1.60 GHz with 8 GB of RAM and the second machine is a Windows 7, 2 Duo @ 2,10 GHz with 2 GB of RAM. The two machines are connected via WiFi.

Figure 7 shows the average response time of each packet that varies in the order of milliseconds. We found this convenient since the cluster heads only exchange messages to synchronize their scans, once it is done the readers start interrogating all tags within range without interruption. Hence, the protocol is not time-consuming and would not affect the readers' waiting time, which is the time for all readers to read the tags. When multiple clusters are triggered, the waiting time does not increase because a cluster head can communicate at a maximum of two adjacent neighbors. Whereas the system

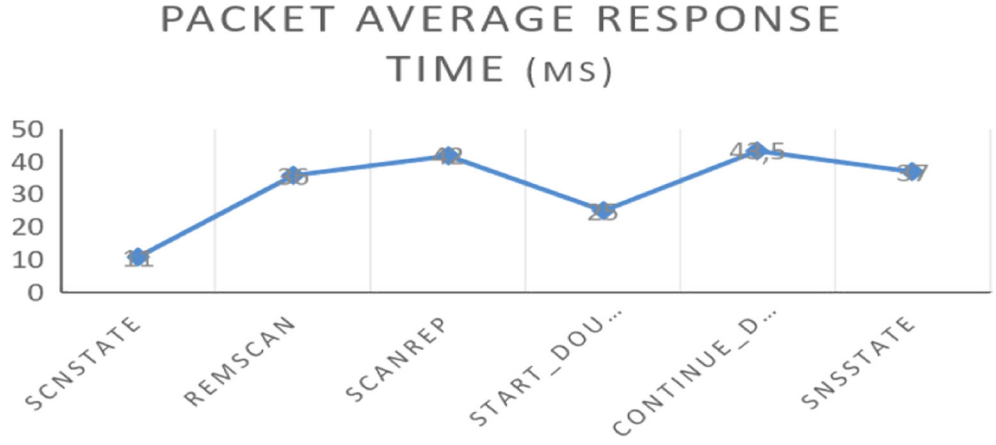

Figure 7: Packet average response time in milliseconds.

throughput continues to increase. As it is not evident to synchronize the time in the order of milliseconds between the two machines we configured the machines as FTP client/server to capture the clock offset, the amount to adjust the local clock to bring it into line with the reference clock.

While the system overhead varies according to the number of common scans as shown in Figure 8. As this number is big as the overhead increases, because after each common scan the cluster heads notify each other to start the next common scan, the maximum packet overhead is 12, which corresponds to four common scans. To determine the overhead in terms of packet size we captured the exchanged packets with Wireshark, each packet is 40 bytes long, the size of the protocol messages is 2 bytes and the other 38 bytes correspond to the UDP, IP, and Ethernet headers. When more cluster heads are triggered, the system overhead will increase relatively to the system throughput.

The SPIN model checker accepts the specification language so-called PROMELA, it is constructed from three basic types of objects, namely processes, data objects, and message channels.

The processes are used to define the behavior, and the message channels are used to model the exchange of data between processes. To build the specification model, we declared a CH process that will be instantiated for the desired number of neighbors, and a reader process just to simulate tag reading. The CH process has three parameters: Its ID, a Boolean variable to present the scan request coming from the system and its default scan sequence. While the reader process has two parameters: Its ID and the ID of its CH. We present here the simulation result of the double scan coordination explained in section B, where the designated CH finds only one neighbor on scanning state. For this, we instantiated two CH processes consecutively, and four reader processes for each CH. Figure 9 illustrates the message sequence chart of the double scan coordination process. To seek simplicity, we omit the “OK” message. Protocol pseudo code is provided in Figure 8.

```
1:   If scan request is received
2:      check the neighbors status
3:   if the remaining scan request is received
4:      send back the scan report message
5:   if end of current scan OR start of double scan OR start of triple scan OR continue scan notification is
received
6:     transmit the scan request to corresponding readers to launch the relative scan sequence
7:    if one scan sequence is done
8:      notify the neighbors of end of current scan
9:   if all scan sequences are done
10:      notify the neighbor of end of scan
- CASE: No Neighbor is on Scanning State
1: notify the neighbors of start of the scan
2: transmit the scan request to corresponding readers to launch the first sequence of scan
- CASE: One Neighbor is on Scanning State
1: ask the neighbor for its remaining scan sequences
2: redefine the scan sequence as to start with the neighbor's remaining scans
3: ask the neighbor to start the synchronized double scan
4: transmit the scan request to corresponding readers to launch the first sequence of scan
- CASE: Two Neighbors are on Scanning State
1: ask the neighbor for their remaining scan sequences
2:   if at least one neighbor has all remaining scans common
3:      redefine necessary scan sequences
4:      send the neighbors modified sequences if any
5:      and ask them to start the synchronized triple scan
6:      transmit the scan request to corresponding readers to launch the first sequence of scan
7:   else if both neighbors have common and uncommon remaining scans
8:      redefine necessary scan sequences
9:      send the neighbors modified sequences if any
10:     and ask them to continue their uncommon scans
```

```
11:       if uncommon scans are done
12:           ask the neighbors to start the synchronized triple scan
13:           transmit the scan request to corresponding readers to launch the first sequence of scan
14:   else if both neighbors have only uncommon remaining scans
15:     redefine necessary scan sequences
16:     ask the neighbors to continue their scans
17:       if the neighbors terminate their scans at the same time
18:           transmit the scan request to corresponding readers to launch the first sequence of scan
19:       else
20:           ask the neighbor to start the synchronized double scan
21:           transmit the scan request to corresponding readers to launch the first sequence of scan
```

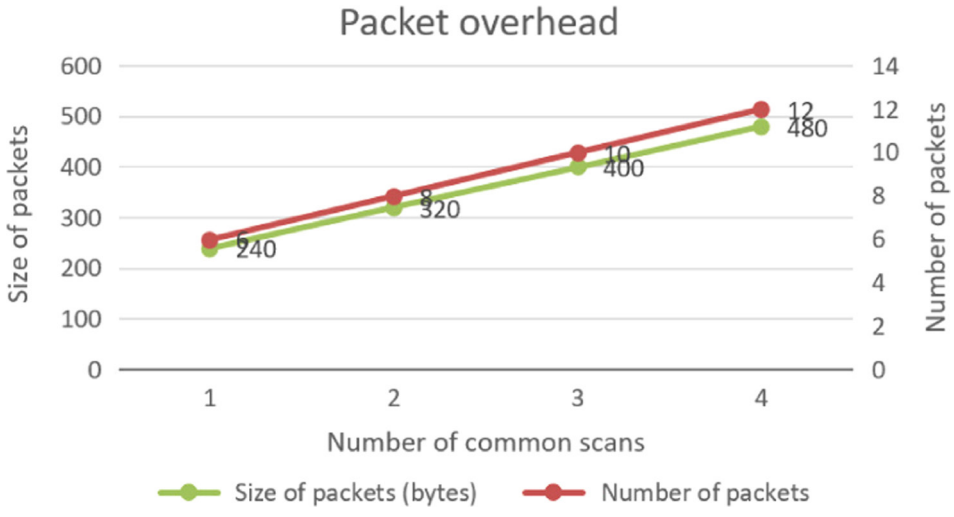

Figure 8: Packet overhead.

Conclusion

In this paper, we proposed a collaborative communication protocol to avoid the reader-to-tag collision problem. To respond to the requirements of an automated stocktaking use case, we particularly addressed its deployed RFID architecture, which is composed of static nodes arranged in a square grid topology. This kind of architecture is opted by many other RFID applications in different industries. The stocktaking process is a crucial operation for efficient supply chain management, hence it cannot be fault-tolerant in terms of the missed tags. On the other hand, existing reader collision avoidance protocols do not guaranty a system free of collisions, none of them has registered any collisions, and no missed tags. Therefore, we proposed a personalized solution of RFID collision avoidance for a stocktaking use case. For that, we combined scheduling and clustering approaches to share the available time slots among the readers in a decentralized mechanism. Where distributed cluster heads in the RFID network are in charge of arranging the readers' activity.

The protocol also provides collaboration among the cluster heads to avoid collisions among readers of adjacent clusters. This is done by synchronizing the operations of the readers belonging to different clusters.

The protocol was validated by Promela/Spin, we presented also a proof of concept using java runtime to have a prior idea about the packets response time and the system overhead. Our future work will consist of full performance analysis by simulating a large RFID network under the R2RIS simulator.

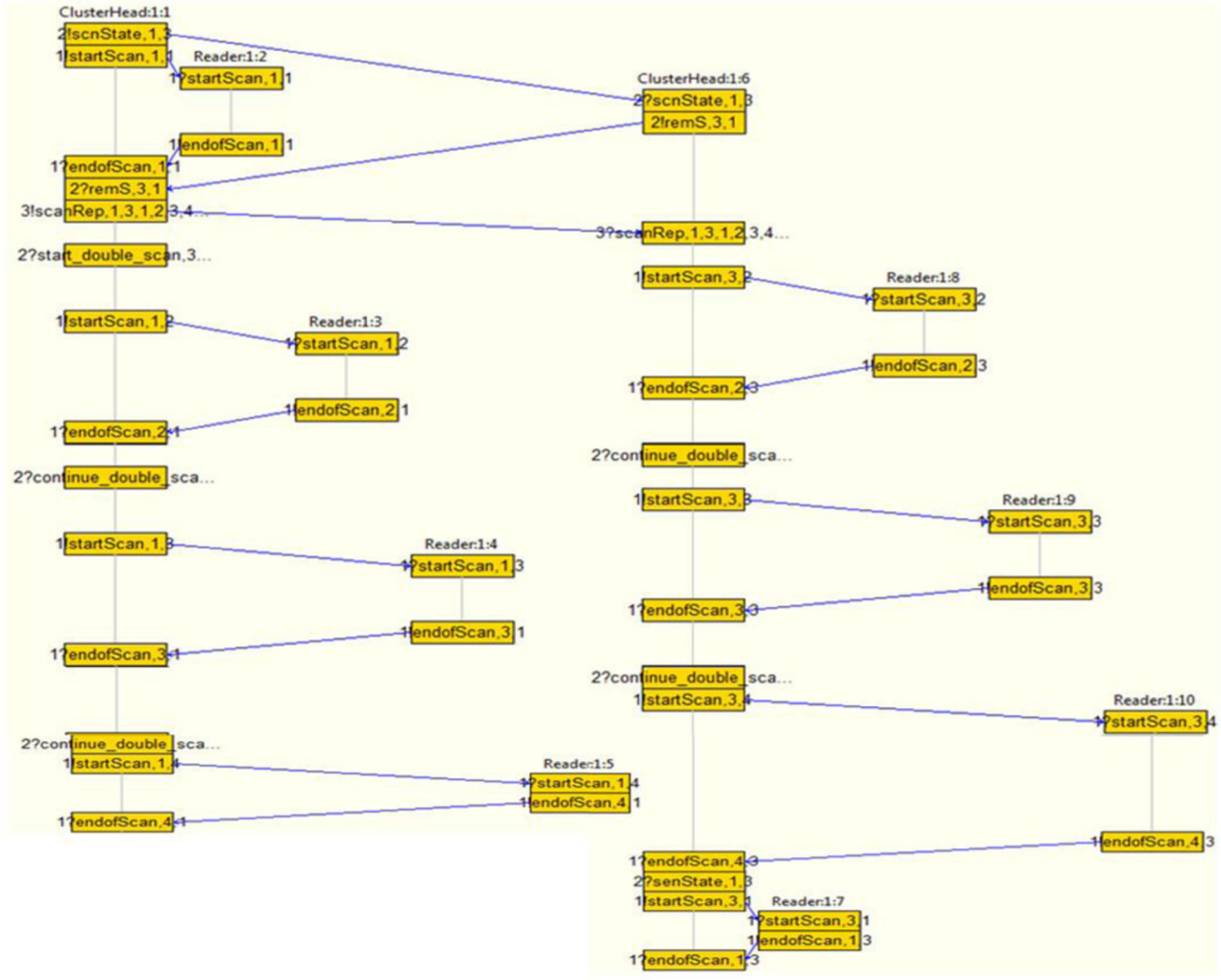

Figure 9: Message sequence chart of double scan coordination.

Acknowledgments

This work was conducted within the research project RSCM2015-2018. We would like to thank the Moroccan MS, MESRSFC, and CNRST for their support.

Literature Cited

Ali, A., Jeff, K. and Amjad, G. 2017. "Tracking and automating a library system using radio frequency identification technology", *International Journal on Smart Sensing and Intelligent Systems* 10(2): 425–450.

Birari, S. M. and Iyer, S. 2005. "PULSE: a MAC protocol for RFID networks", In Enokido, T., Yan, L., Xiao, B., Kim, D., Dai, Y. and Yang, L. T. (Eds), *International Conference on Embedded and Ubiquitous Computing*, Springer, Berling and Heidelberg, pp. 1036–1046.

Dai, H., Lai, S. and Zhu, H. 2007. A multi-channel MAC protocol for RFID reader networks In *International Conference on Wireless Communications, Networking and Mobile Computing 2007, WiCom*, IEEE, Shanghai, pp. 2093–2096.

Engels, D. W. 2002. "White paper: The Reader Collision Problem", Auto-id Center Massachusetts Institute of Technology.

Engels, D. W. and Sarma, S. E. 2002. "The reader collision problem", *International Conference on Systems, Man and Cybernetics, 2002*, IEEE, Yasmine Hammamet 3, 6 pp.

Eom, J. B., Yim, S. B. and Lee, T. J. 2009. "An efficient reader anticollision algorithm in dense RFID networks with mobile RFID readers", *IEEE Transactions on Industrial Electronics* 56(7): 2326–2336.

Finkenzeller, K. 2010. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-field Communication* 3rd ed., John Wiley & Sons, Munich.

Golsorkhtabaramiri, M. and Issazadehkojidi, N. 2017. "A distance based RFID reader collision avoidance protocol for dense reader environments", *Wireless Personal Communications* 95(2): 1781–1798.

Golsorkhtabaramiri, M., Hosseinzadeh, M., Reshadi, M. and Rahmani, A. M. 2015. "A reader anti-collision protocol for RFID-enhanced wireless sensor networks", *Wireless Personal Communications* 81(2): 893–905.

Hwang, K. I., Kim, K. T. and Eom, D. S. 2006. "DiCa: distributed tag access with collision-avoidance among mobile RFID readers", In Zhou, X. et al. (Eds), *International Conference on Embedded and Ubiquitous Computing 2006, EUC*, Vol. 4097, Springer, Berlin and Heidelberg, pp. 413–422.

Jiang, Y., Zhang, R., Cheng, W. and Sun, W. 2016. "An efficient multi-channel reader collision avoidance protocol in RFID systems", *International Conference on Wireless Communications and Networking*, IEEE, pp. 1–6.

Joshi, G. P. and Kim, S. W. 2008. "Survey, nomenclature and comparison of reader anti-collision protocols", *RFID IETE Technical Review* 25(5): 234–243.

Krebs, D. and Liard, M. J. 2001. White paper: global markets and applications for radio frequency identification Venture Development Corporation.

Mbacke, A. A., Mitton, N. and Rivano, H. 2018. "A survey of RFID readers anti-collision protocols", *IEEE Journal of Radio Frequency Identification* 2(1): 38–48.

Mekala, S., Thanagaraj, M., Chandranath, M. and Vasanta Kumaran, K. K. 2017. "An Intelligence Super Mart Billing System", *International Journal on Smart Sensing and Intelligent Systems* 10(5): 414–425.

Nagpurkar, A. W. and Jaiswal, S. K. 2015. "An overview of WSN and RFID network integration", *International Conference on Electronics and Communication Systems, ICECS*, IEEE, Coimbatore, pp. 497–502.

Rezaie, H. and Golsorkhtabaramiri, M. 2018. "A fair reader collision avoidance protocol for RFID dense reader environments", *Wireless Networks* 24(6): 1953–1964.

Safa, H., El-Hajj, W. and Meguerditchian, C. 2015. "A distributed multi-channel reader anti-collision algorithm for RFID environments", *Computer Communications* 64: 44–56.

Waldrop, J., Engels, D. W. and Sarma, S. E. 2003. "Colorwave: A MAC for RFID reader networks", *International Conference on Wireless Communications and Networking 2003, WCNC*, IEEE, New Orleans, LA 3: 1701–1704.