2005

Redundant-Reader Elimination in RFID Systems

Bogdan Carbunar

Murali Krishna Ramanathan

Mehmet Koyuturk

Christoph M. Hoffmann
Purdue University, cmh@cs.purdue.edu

Ananth Y. Grama
Purdue University, ayg@cs.purdue.edu

Report Number:
05-013

REDUNDANT-READER ELIMINATION IN RFID SYSTEMS

Bogdan Carbunar
Murali Krishna Ramanathan
Mehmet Koyuturk
Christoph M. Hoffmann
Ananth Y. Grama

CSD TR #05-013
June 2005

Redundant-Reader Elimination in RFID Systems

Bogdan Carbunar, Murali Krishna Ramanathan, Mehmet Koyuturk, Christoph Hoffmann, Ananth Grama
Department of Computer Science
Purdue University, West Lafayette, IN
Email:{carbunar,rmk,koyuturk,cmh,ayg}@cs.purdue.edu

Abstract— **While recent technological advances have motivated large-scale deployment of RFID systems, a number of critical design issues remain unresolved. In this paper we address two important problems associated with RFIDs. The first problem deals with detecting redundant RFID readers (the *redundant-reader problem*). A related second problem is one of accurately detecting RFID tags, in the presence of reader interference (the *reader collision avoidance* problem). The underlying difficulty associated with these problems arises from the lack of collision detection mechanisms, the potential inability of RFID readers to relay packets generated by other readers, and severe resource constraints on RFID tags. We present a randomized, distributed, and localized solution, RCA, to the reader collision problem. We prove that an optimal solution to the redundant-reader problem is NP-hard and propose a randomized, distributed, and localized approximation algorithm, RRE. We provide a detailed probabilistic analysis of the accuracy and time complexity of RCA and RRE and conduct elaborate simulations to demonstrate their correctness and efficiency.**

I. Introduction

Radio Frequency Identifier (RFID) systems consist of two types of components, RFID transponders (tags) and RFID transceivers (readers). RFID tags are comprised of a small integrated circuit for storing information and an antenna used for communication. Tags may be passive, i.e., they do not require batteries and instead use energy of the received signal to reveal its stored information. RFID readers are capable of reading the information stored at non line-of-sight RFID tags placed in their vicinity and communicate it through a wired or wireless interface to a central database. Supply chain automation, cold chain management (temperature logging), identification of products at check-out points, access control and security, are among common applications of RFID systems.

Significant investment by major retailers such as Wal-Mart and Tesco, mandating their manufacturers to place tags on cases and pallets provides a strong motivation for the large scale deployment of RFID systems. This investment is based on recent technological advances that have made possible, mass production of inexpensive RFID tags. Their cost is expected to drop below the 5 cents/tag threshold [1]. The main advantages of RFID systems are price efficiency (billions of dollars in anticipated savings for Wal-Mart alone [2]) and accuracy of stock management (GAP documented an increase of accuracy from 85% to 99.9% when using RFID technology [3]).

The miniaturization of RFID readers (SkyeRead M1-Mini [4]), coupled with their enhancement with Wi-Fi or cellular capabilities (SmartCode [5]), broadens the scope of applications of RFID systems. Wireless RFID systems, similar to wireless sensor networks, can be deployed on-line instead of being statically pre-installed. Unlike sensor networks, wireless RFID systems have the ability to decouple the sensing and communication functions. Since RFID tags interfaceable with external sensors, such as temperature and shock sensors or tamper indicators, have already been produced [6], wireless RFID systems can be easily extended with new sensing capabilities by deploying corresponding RFID tag types. Furthermore, the existing compatibility between recent RFID readers (SkyeRead M1-Mini [4]) and MICA2DOT motes motivates integration of wireless sensor and RFID networks. Such a hybrid infrastructure combines the affordability of deployment with the efficient and accurate identification and monitoring of objects.

The main problem addressed in this paper, of extending the lifetime of wireless RFID reader networks, stems from the limited battery life of wireless RFID readers and the need for accurate monitoring of areas of interest. This, in turn requires dense deployment of wireless RFID tags and readers. The solution proposed in this paper is based on the identification of redundant RFID readers, which we define in terms of the covered RFID tags. The temporary deactivation of such readers does not reduce the number of tags covered by the initial reader network. Our purpose is to detect the maximum number of redundant readers that can be safely turned off simultaneously. For example, in Figure 1, all RFID readers are redundant (i.e., each tag is covered by multiple readers), however, only a subset may be simultaneously deactivated.

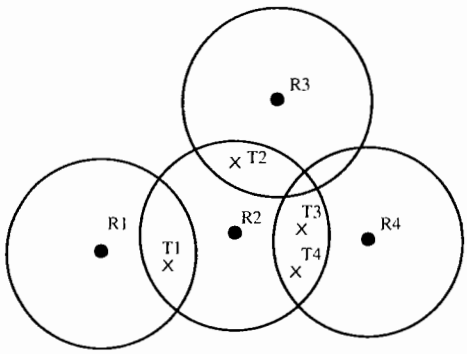


Fig. 1. Redundant reader example: readers R_1, R_2, R_3 and R_4 are redundant since the tags covered by each is covered by at least one other reader. This redundancy information would not be detected by a sensor redundancy detection algorithm, since the coverage areas of any of the readers are not subsumed by the others. The optimal solution requires only R_2 to be active, while the other readers may be turned off.

While the problem of determining coverage redundancy has been extensively studied in wireless sensor networks [7], [8], [9], [10], it differs from the redundant RFID reader elimination problem in several aspects. First, coverage is defined in terms of contiguous circular areas associated with sensors, whereas in RFID systems coverage is defined in terms of discrete points (RFID tags). Second, solution to this problem for sensor networks relies on the existence of location information, or at least the ability to estimate distances between adjacent sensors. Due to the limited resources of RFID tags, in RFID systems such an assumption in not reasonable. Third, the limited resources of RFID tags coupled with the potential inability of RFID readers to act as packet routers, considerably restricts the solution space of the redundant-reader problem.

We prove that even with centralized knowledge of the RFID system topology, an optimal solution for the redundant-reader elimination problem is NP-hard. We introduce a randomized, decentralized, and localized approximation algorithm for the redundant-reader elimination problem, called RRE. For each reader, the first step of RRE detects the set of RFID tags placed in the vicinity of a reader. The difficulty associated with this step rests on the potential occurrence of reader collisions at tags. Reader collisions occur at tags situated in the vicinity of two or more readers that are simultaneously sending queries. Such tags may be unable to correctly decode the queries, potentially leading to unexpected behavior. The absence of global topology information, where readers might not be aware of generated collisions, makes the task of accurate query scheduling difficult. We propose a randomized, distributed, and localized algorithm, RCA, for avoiding reader collisions and allowing RFID readers to accurately detect the tags in their vicinity.

In the second step of RRE, each RFID reader attempts to write its tag count (number of covered tags) on to all its covered tags. A tag placed in the vicinity of several readers will overwrite the count stored on behalf of a reader only if the new value is larger. The reader that issued the highest count for a tag, *locks* the tag. In the final step of RRE, each reader sequentially queries all its covered tags to discover the ones it has locked. A reader that has not locked any of its covered tags is declared redundant.

RCA and the subsequent steps of RRE rely on a randomized querying technique, for avoiding reader collisions. Section III describes this technique in detail in the context of RCA. Section IV defines the redundant-reader problem and proves its NP-hardness and Section V presents our solution, RRE. Section VI presents our simulation results and Section VIII draws conclusions.

II. Network Model

Our algorithms are designed under the following conservative assumptions. Any relaxation of these conditions will only improve the performance of our algorithms.

- Our algorithms are applicable to any number of RFID readers and tags and we make no assumptions on the underlying reader or tag topology. We do not assume the presence of a centralized entity capable of collecting the topology of the reader network or controlling the behavior of individual readers. Thus, our algorithms do not rely on the ability of RFID readers to communicate.
- We assume the presence of passive tags only, as opposed to active tags (the latter are more powerful and expensive). Therefore, RFID tags use the energy of the received signal in order to answer queries from readers.
- Tags have limited memory. Part of it is read-only, used to store unique identifiers, and part of it is writable. Tags are capable of doing prefix matching.
- RFID readers are able to detect RFID tag collisions, occurring when multiple RFID tags reply to the same query.

III. Reader Collision Avoidance

We first examine the impact of collisions, an essential aspect of RFID systems. More precisely, we look at a popular solution for tag collisions and then propose an efficient solution for avoiding reader collisions.

Tag Collisions: The area around an RFID reader, where RFID tags can receive the reader's signal and their replies can be correctly decoded by the reader, is called the interrogation zone of the reader. The main functionality of an RFID reader is to detect the unique identifiers of all the RFID tags in its interrogation zone. Simultaneous replies from RFID tags situated in the interrogation zone of a reader make accurate decoding of signals impossible. This problem, known as the tag-collision problem, prevents an RFID reader from simultaneously reading all its covered RFID tags.

Several techniques have been proposed to solve the tag-collision problem. A popular solution, known as the tree walking algorithm [11](TWA), is based on a recursive traversal of the binary name tree of RFID tag identifiers. The reader initially sends a broadcast query containing the "0" string. All RFID tags in its interrogation zone whose id prefix is "0" must reply. If a reply is received, or a tag-collision is detected, the reader recurses on the left and then the right subtree of "0", rooted at "00" and "01". However, if no reply is received, the reader concludes the absence of "0"-prefixed tags in its interrogation zone and subsequently sends a "1" query. For a reader, the complexity of TWA is proportional to the number of tags present in its interrogation zone and to the length of the binary representation of RFID tag identifiers.

Reader Collisions: TWA [11] does not solve the following related problem. When two RFID readers are placed close enough for their interrogation zones to overlap but far enough to prevent direct communication, RFID tags placed within the intersection area of the interrogation zones may receive queries from both readers simultaneously. Such queries, potentially part of the TWA protocol, will interfere, preventing the corresponding RFID tags from correctly decoding the queries. These tags may escape detection by any reader in the system.

Outline of RCA: We propose a randomized, distributed and localized solution to the reader collision problem. Our algorithm, named RCA (Reader Collision Avoidance), is presented in the context of TWA. However, a similar approach can be extended to any scenario where a reader needs to communicate with a tag. Similar to TWA, in RCA an RFID reader sends a broadcast query containing a certain prefix expected to match the identifiers of RFID tags in its interrogation zone. However, unlike TWA, where the lack of an answer is considered to denote absence of matching RFID tags, RCA backs-off for a random number of time frames and repeats the query. The purpose of the random back-off and query repetition is to ensure w.h.p. the choice of a time frame not picked by another RFID reader, thus avoiding reader collisions.

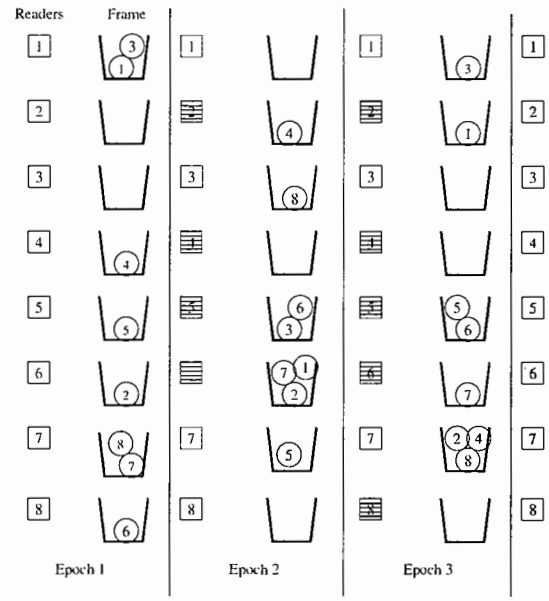


Fig. 2. Illustration of our solution – squares denote readers, buckets denote frames and labeled balls (the label indicating the source of the ball) denote queries sent by readers during different time frames. In each epoch, readers that have successfully transmitted information to a tag are shown shaded. A reader is successful if its corresponding ball is the only one present in a bin, i.e., it did not collide with any other reader. After the first epoch, readers 2, 4, 5, and 6 are successful since their corresponding balls uniquely occupy their respective buckets. During the second epoch, reader 8 also succeeds. Eventually (see Section **??** for a bound on the number of epochs) all readers are successful.

The premise of the algorithm is as follows. An RFID reader divides time into disjoint epochs and each epoch is further divided into multiple disjoint time frames. In each epoch, an RFID reader picks a frame uniformly at random and sends its query in that frame. If no tag answer is received, the RFID reader repeats the query in a randomly chosen time frame of the next epoch. If a reader collision at matching RFID tags has occurred during the query, the query duplication correlated with the random backoff decreases the chances of repeated reader collisions. Section III-A proves that if a query is not answered $O(\log \psi)$ times, where ψ is the total number of RFID readers, w.h.p., there are no RFID tags matching the query in the interrogation zone of the RFID reader. If, however, an answer is received, either as a clear tag response or by detecting a tag collision, the RFID reader recursively moves to the next query, as in the TWA algorithm.

The choice of repeating a query $O(\log \psi)$ times is made under the conservative assumption that all RFID readers interfere with each other at all tags. Hence,

Algorithm 1 The generic reader and tag behavior. getRandom(v_1, v_2) returns a random integer value between v_1 and v_2 and bCast(packet) is used to broadcast packet.

```
1.Object implementation RFIDTag;
2.  T_id : integer; #tag identifier
3.  inQ : queue; #queue of incoming packets
4.  Operation run()
5.      guard inQ.first.type = query do
6.          if prefixMatch(inQ.first.tid, T_id) then
7.              bCast(new packet(TAG));
8.          fi
9.      od
10. end
11. Object implementation RFIDReader;
12. count, e : integer; #epochs per bit read
13. frame, n : integer; #time frames in each epoch
14. T, T_out : integer; #time out value
15. inQ : queue; #queue of incoming packets
16. Operation treeWalk(prefix : integer)
17.     count := 0;
18.     while count + + < e do
19.         frame := getRandom(0, n);
20.         sleep(frame);
21.         T = getTime();
22.         bCast(new packet(query, prefix));
23.         guard inQ.first.type = TAG_COL || TAG do
24.             treeWalk(prefix + "0");
25.             treeWalk(prefix + "1");
26.         od
27.         guard getTime() − T ≥ T_out do
28.             sleep(n − frame − 1);
29.         od
30.     od
31. end
```

the bound that we provide is the worst case bound. However, this is not always the case. In our experiments (see Section VI), we show that in realistic scenarios of random deployment of RFID readers and tags, much fewer repetitions are needed in order to allow RFID readers to accurately detect RFID tags.

Implementation: Algorithm 1 presents the pseudocode for RCA using an Orca [12] like syntax. Orca is a parallel programming language for distributed systems, that provides elegant constructs for expressing reactive behavior, such as *guards*. Operations consist of one or more guards with syntax

guard expression **do** statementSeq **od**,

where expression is a boolean expression and statementSeq is a sequence of statements. The operation containing guards blocks until one or more guards are true. Then one of the satisfied guards is randomly chosen and its statements are executed atomically.

The operation of a tag is shown in Algorithm 1, lines 1-10. A tag replies only to queries containing strings whose prefixes match its own identifier (lines 5-9). inQ.first is used to denote the packet currently received by the tag. The operation of a reader is shown in Algorithm 1, lines 11-31. Time is divided into epochs, with each epoch containing a fixed number, n, of time frames. The duration of a time frame is equal to the time necessary for a query to propagate from a reader to a tag. For each prefix queried, the reader waits for a maximum of e epochs (line 18) and in each epoch sends exactly one broadcast message containing the prefix. During each epoch, the broadcast message is sent in a randomly chosen time frame (lines 19-22).

The lack of a reply may denote either the absence of a tag matching the queried prefix in the interrogation zone, or the occurrence of reader collisions at such tags. If less than e queries with the current prefix have been sent, the reader waits until the beginning of the next epoch to repeat the above process (lines 27-29). If no reply or collision is detected after e rounds, the reader ignores the subtree rooted at the queried prefix. However, the receipt of an individual reply or the detection of a tag collision stops this process, since the reader can now safely recurse on the two children of the employed prefix (lines 23-26).

A. Analysis

We present an analysis of RCA based on two fundamental abstractions in randomized algorithms, *viz.* the coupon collector abstraction and a balls and bins paradigm. For the sake of completeness, we define the coupon collector problem as in Motwani and Raghavan [13].

Coupon-Collector: Given a set of coupons containing n unique coupon types, the number of samples required to obtain w.h.p. a coupon of each type, is nH_n, where H_n is $\mathtt{O}(\log \mathtt{n})$.

Let ψ be the total number of readers and γ the total number of RFID tags in the system, τ be the number of time frames per epoch and β be the bit size of RFID tag identifiers. Our first goal is to evaluate the number of epochs per query, x, required to ensure the success of the query. To establish an upper bound, we assume a star topology in which interrogation zones of all ψ RFID readers share all τ RFID tags. Note that this is a worst case assumption. Each frame is considered to be a bin and a query of an RFID reader is modeled as a ball. We first prove the following lemma.

Lemma 1: In each epoch of the RCA process, the expected number of readers that send a message without

a collision is $\psi e^{-\frac{\psi}{\tau}}$.

Proof: When ψ readers send a message uniformly at random in any one of the τ frames of an epoch, the distribution of the messages in each frame follows a Poisson distribution [13]. Therefore, if X_i is a random variable that is equal to the number of messages sent by different readers in frame i, the probability of exactly one message being sent in frame i is given by

$$P(X_i = 1) = \frac{\psi}{\tau} e^{-\frac{\psi}{\tau}}$$

Since there are τ frames, the average number of frames where exactly one message is sent is $\psi e^{-\frac{\psi}{\tau}}$. ■

Using the coupon collector paradigm we can prove the following lemma.

Lemma 2: The RCA process is dominated by the coupon collector process.

Proof: In RCA, an RFID reader sends a query until the upper bound, x, is reached. The approach can be modeled as a coupon collector process, where each reader is a coupon type. A coupon type is chosen if the query sent by the corresponding reader during its chosen time frame of the current epoch is the only query sent by a reader during the same time frame. From Lemma 1, on average $\psi e^{-\frac{\psi}{\tau}}$ coupon types are selected during each frame. This is similar to choosing $\psi e^{-\frac{\psi}{\tau}}$ coupons (of the coupon collector process) and then placing back the chosen coupons into the set, instead of choosing a single coupon and replacing it immediately. This increases the rate at which the coupon types are chosen. Thus, the number of epochs needed for each RFID reader to send the only query during a time frame is at most the number of samplings in the actual coupon collector process. ■

We can now prove the following theorem, providing an upper bound on the number of query repetitions in RCA.

Theorem 1: Setting the number of time frames per epoch, τ, to be the total number of readers, ψ, in RCA, requires only $O(\log \psi)$ query repetitions to ensure w.h.p. the receipt of a reader's query by the target RFID tags in its interrogation zone.

Proof: If x is the number of query repetitions, using Lemma 1 and Lemma 2, we get

$$x \ \psi \ e^{-\frac{\psi}{\tau}} \ \leq \ c\psi \log \psi.$$

When $\tau = \psi$, $x = O(\log \psi)$. ■

We can now provide the worst case time complexity of RCA.

Complexity of RCA: The time complexity of RCA, T_{RCA} is $O(\gamma \log \beta \log \psi)$ time epochs.

Proof: Since each reader covers γ tags of bit size β, the number of query types is $O(\gamma \log \beta)$. Theorem 1 completes the proof. ■

B. Discussion

Synchronization.: An important observation is that time synchronization is not required from distinct readers. The starting points of time epochs can be maintained locally by all readers, independent of other readers and tags. The analysis given above is for the worst case scenario, where all readers are assumed to have a query to send during the current epoch. The asynchrony of readers can imply fewer collisions which in turn determine an earlier reception of positive acknowledgments from the tags. Thus, our worst case bounds presented above hold for an asynchronous system.

Star topology.: For ease of analysis, we assumed a star topology consisting of ψ readers and one tag. This is also a worst case scenario. In practice, not all interrogation zones of readers overlap at tags. Hence, any other topology would only improve performance of the system i.e., fewer collisions and an earlier reception of acknowledgments from matching tags. Furthermore, the above analysis was performed with one tag and multiple readers. As the number of tags placed inside the interrogation zone of readers increases, so does the chance of multiple tags to match the current query. This implies that fewer query repetitions will be required, since the reader needs to receive only one tag answer in order to proceed with the next query and different tags can be shared with different readers.

Independent operation.: Our analysis is independent of the current stage of the tree walking algorithm of each reader. For example, a reader may broadcast a "0" query while another reader broadcasts a prefix "10". This can occur not only because of the asynchrony assumption, but also because different readers may cover different tags. Our analysis is independent on the content of the queries, by abstracting queries sent at certain time frames as balls being thrown into bins.

Exponential back-off.: A potential improvement of RCA could seem to be using exponentially increasing epoch sizes instead of constant sized epochs. In the case of exponential back-off, the number of frames in

each epoch for a query is equal to 2^i, where i is the current number of repetitions for the given query. Then, following a reasoning similar to the one employed for Theorem 1, the number of repetitions per query, x, is given by the solution to the following equation, $\sum_{i=0}^{x} e^{-\frac{\psi}{2^i}} = \log \psi$. While for large values of ψ, the convergence of a solution employing exponential back-off times is faster than its constant counterpart, it is obvious that the number of time frames also increases exponentially. This observation along with the knowledge of the number of readers motivates our use of constant epoch sizes, instead of exponential epoch sizes generally used in Ethernet and other exponential back-off approaches[1].

IV. The Redundant-Reader Problem

We now define the redundant RFID reader problem and prove that finding the optimal solution is NP-hard. We define a redundant reader as follows:

Definition 1: A redundant RFID reader covers a set of RFID tags that are also covered by other RFID readers.

According to this definition, all the RFID readers in Figure 1 are redundant. A simple solution to detect the redundant RFID readers is to have all RFID readers simultaneously broadcast a query containing the empty string. Since all the RFID tags that receive such a query must answer, an RFID reader that receives no reply is redundant. This is either because the RFID reader covers no RFID tag, or because interference occurred at all its covered RFID tags. Such a solution has two important drawbacks. First, it requires time synchronization between all RFID readers. Second, turning off all the redundant RFID readers may leave uncovered tags that were previously covered by at least two redundant readers (blind tags). For example, in Figure 1, the simultaneous deactivation of R_1 and R_2 leaves RFID tag T_1 uncovered.

In order to maximize the number of RFID readers that can be simultaneously deactivated, the minimum number of readers that cover all RFID tags needs to be discovered. We define the redundant-reader problem as follows:

[1]In Ethernet, the number of entities sending a message decreases in subsequent epochs, due to the presence of collision detection mechanisms.

Redundant-Reader Problem: Given a set of RFID tags and a set of RFID readers covering all the RFID tags, find the minimum cardinality subset of RFID readers, covering all the tags.

For example, in Figure 1, R_2 is the only reader that needs to be active. In order to prove that the redundant-reader problem is NP-hard, we first prove the following lemma, illustrated in Figure 3.

Lemma 3: Given a set of n points, $p_1, p_2, .., p_n$, placed inside a circle of radius R, there exists a subset of 3 of the n points, p_i, p_j, p_k, such that all the n points are placed inside $C(O_{ijk}, R)$. O_{ijk} is the mass center of p_i, p_j, p_k and $C(x, r)$ denotes the circle centered at x with radius R.

Proof: We provide a constructive proof. If all the points are covered by a circle of radius R, then a circle of radius R going through 2 of the points and covering all the other points exists (see Figure 3). If the circle has a third of the n points on its perimeter, then we have completed the proof. Otherwise, shrink the circle until its perimeter touches a third point. The resulting circle has radius less than or equal to R, is the circumcircle of three of the n points, and covers all the other points. ■

We can now prove the following important result.

Theorem 2: The redundant-reader problem is NP-hard.

Proof: By reduction from the geometric disk cover (DC) problem, known to be NP-hard [14]. The input for the DC problem consists of a set of m points and a value R. The output consists of the minimum number of disks of radius R that cover all the points. We use the following polynomial-time reduction from DC to the redundant-reader problem. Add a disk of radius R centered at each point in the input set of DC. Then, for all combinations of 3 points of the input set of DC, add a disk of radius R, centered at the mass center of the 3 points. Let S denote the set of all disks created. It is clear that the disks in S cover all the input points of DC. Moreover, as a direct consequence of Lemma 3, the disks that form the solution for the DC problem are contained in S. The reduction has $O(m^3)$ complexity. If a polynomial time algorithm for the redundant-reader problem would exist, we could find the minimum number of disks needed to cover the points, which cannot be worse than the solution for the DC problem, in polynomial time. ■

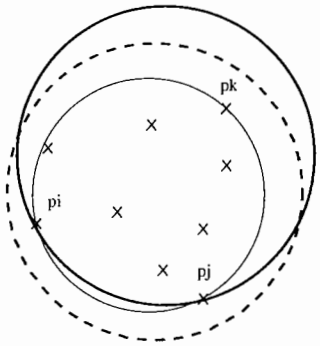


Fig. 3. Set of points covered by a circle of radius R, shown with an interrupted perimeter. There is a circle of radius R going through points p_i and p_j and covering all the other points. Shrink this circle until it first touches one more point, p_k. The resulting circle, has radius less than or equal to R.

V. Redundant Reader Elimination Algorithm

We propose a randomized, distributed and localized approximation algorithm for the redundant-reader problem. As specified in Section II, we make no assumption on the topology of the RFID reader network, effectively claiming no direct communication between RFID readers. We assume, however, the existence of writable tags that are able to store information upon requests from in-range RFID readers. We assume initially that RCA (see Section III) has been previously executed by all readers to identify RFID tags in their vicinity. Later in this section we discuss a simple modification to our algorithm to remove this assumption.

Outline of RRE: RRE (Redundant-Reader Elimination) consists of two steps. In the first step, each RFID reader attempts to write its tag count (number of covered tags) to all its covered RFID tags. An RFID tag only stores the highest value seen, along with the identity of the corresponding reader. For this, each reader issues a write command containing its reader identifier and tag count. Similar to RCA, the write operation is performed during $O(\log \psi)$ consecutive epochs, once per epoch. During each epoch, the time frame for sending the write request is randomly chosen. As shown in Section III-A, this process ensures w.h.p. that at least one write command issued by each RFID reader will be correctly received by all its covered RFID tags. Thus, after $O(\log \psi)$ epochs, each RFID tag stores the largest number of tags covered by an RFID reader situated in its vicinity, along with the identity of that reader, called *holder* of the tag.

In the second step, an RFID reader queries each of its covered RFID tags and reads the identity of the tag's holder. A reader that locked at least one tag is responsible for monitoring the tag and will have to remain active.

Algorithm 2 The generic RFID reader and writable tag behavior for detecting redundant readers.

```
1.Object implementation WritableRFIDTag;
2.  R_id : integer; #identifier of locking reader
3.  count = 0 : integer; #count of highest bidder
4.  Operation run()
5.      guard inQ.first.type = write do
6.          if inQ.first.c > count then
7.              R_id := inQ.first.rid;
8.              count := inQ.first.c;
9.          fi;
10.     guard inQ.first.type = read do
11.         bCast(new packet(T_id, R_id, count));
12.     od
13. end

14.Object implementation RFIDReader;
15. R_id : integer]; #reader identifier
16. tags : array[integer] of integer; #covered tags
17. redundant = true : boolean;
18. Operation isRedundant(prefix : integer)
19.     while count + + < e do
20.         frame := getRandom(0, n);
21.         sleep(frame);
22.         bCast(new packet(write, R_id, tags.size));
23.         sleep(n − frame − 1);
24.     od
25.     for i in 1..tags.size do
26.         while count + + < e do
27.             T = getTime();
28.             frame := getRandom(0, n);
29.             sleep(frame);
30.             bCast(new packet(read, tags[i]));
31.             guard inQ.first.tid = tags[i] do
32.                 if inQ.rid ! = R_id then
33.                   redundant := false;
34.                 od
35.             guard getTime() − T > n do od
36.         od
37.     od
38.     if redundant = true do turnOff(); fi
39. end
```

However, a reader that has locked no tag can be safely turned off. This is because all the tags covered by that reader are already covered by other readers that will stay active. The read queries issued by a reader for each of its tags are similarly repeated during random time frames for $O(\log \psi)$ time epochs to avoid reader collisions occurring at queried tags.

Implementation: Algorithm 2 illustrates our solution, which assumes writable RFID tags. The functionality of a writable tag is shown in operation `run` of `WritableRFIDTag` (lines 4-13). The RFID reader and tag objects inherit the corresponding variables defined in Algorithm 1. When a writable tag receives a `write` command containing the identifier of the reader issuing the command and its tag count, it saves the values locally

only if the tag count is larger than the value currently stored. When the command received is a `read`, the tag returns a packet containing its identifier followed by the reader's identifier and count value stored locally.

The detection of redundant RFID readers is exhibited in operation `isRedundant` of `RFIDReader` (lines 18-39). First, a reader selects a random time frame during `e` consecutive epochs, and broadcasts a `write` command containing its identifier and tag count (lines 19-24). Subsequently, it queries each of its covered tags, using a `read` command, for `e` consecutive time epochs in order to find the tag's holder (lines 25-37). Note that after sending a `read` command, at the chosen time frame, the reader waits either to receive a reply from the queried tag or for the epoch to end (lines 31-35).

A. Discussion

Synchronization: We have assumed until now that all RFID readers have already executed RCA, detecting all the RFID tags in their interrogation zone. This assumption ensures that on completion of the first step of RRE, tags placed in the vicinity of at least two readers store the highest number of tags covered by the readers. For example, in Figure 1, the count of tag T_3 is 4, from reader R_2. However, if we assume that initially RFID readers are not aware of the identity of adjacent tags and RCA needs to be executed just before RRE, the following scenario may occur (see Figure 1 for illustration): since R_4 only covers two RFID tags, whereas R_2 covers four, R_4 will complete RCA before R_2 and also the first step of RRE. Then, R_4, upon discovering to be the holder of T_3 and T_4, will also decide to stay active, despite its redundancy.

In order to solve this problem, we require active RFID readers to maintain a list of locked tags and to passively listen for RFID tag responses to queries initiated by other readers. When an RFID reader, `R`, receives such a message, of format R_x, T_y, c (see Algorithm 2 line 11), indicating that the holder of tag T_y is R_x with a tag count `c`, if `c` is larger than its own tag count, the reader `R` removes tag T_y from its list of locked tags. When the list is empty, the reader becomes redundant and can be safely turned off. Theorem 1 (see Section III-A) proves that if such a scenario occurs, a reply of content R_x, T_y, c *will* be received by `R` for all tags T_y covered by readers with a larger tag count. Using the example in Figure 1, if R_4 has T_3 and T_4 in its list of locked tags on completion of its first step of RRE, during R_2's execution of the first step of RRE, R_2 will choose at least one time frame during `e` epochs, both for T_3 and T_4, when no other RFID reader is transmitting. Thus, R_4 will overhear the replies of T_3 and T_4. Note that their replies will not generate a tag collision at R_4, since the tags are queried sequentially by R_2 (Algorithm 2 line 30).

System Adaptivity: The current description of RRE assumes a static environment. However, in reality, RFID tags and readers may fail and new components may be randomly deployed. Scenarios where new RFID tags are deployed in areas covered only by inactive readers, or when active RFID readers fail, leaving tags covered only by inactive readers, are particularly important. We present a simple extension of RRE that maintains the invariant of having at least one active RFID reader for each covered tag, in these two scenarios. Our solution periodically re-activates inactive readers and executes RRE on all the readers. Then, the following problem, illustrated in Figure 1, may occur. If the only active reader, R_2, fails when R_1, R_3, and R_4 are re-activated, tags $T_1, .., T_4$ have the associated count value set to 4. The re-activated readers discover, this time inaccurately, their redundancy and switch off, leaving all the tags uncovered.

One solution to this problem executes RCA periodically every `T` time units, to identify all its covered tags, including newly deployed ones. Subsequently, the readers reset the count value of each of their covered tags and re-execute RRE. An RFID tag will agree to set its counter to a smaller value, 0, since 0 is a control value (an RFID reader covering no tags will not issue a `write` command containing a 0 tag count field). Of course, this can lead to a situation in which even though no reader has failed, R_2 sets the counter of its tags to 0 and then to 4, followed by the activation of R_4, R_4's setting the counter of its tags to 0 and then to 2. Then, R_4 and R_2 will both decide to stay active even though R_4 is redundant. A solution for this scenario is to set the period `T` of each reader to be inversely proportional to the tag count of the reader. Then, R_2 will execute this procedure more often than R_4, eventually causing R_4 to discover its redundancy. Another solution, requiring more complex tags, is to have timers on tags. A tag may store a tag count only for a limited time, until the expiration of its timer. The timer is set when a new tag count value is stored by the tag.

B. Analysis

Since the number of RFID tags covered by a reader is not known before running RCA, accurately evaluating the time necessary for RCA to complete is difficult. Even though the duration of the first step of RRE is fixed, $\log \psi$ time epochs, the second step of RRE may

start at different times even for readers that have started RCA simultaneously. The question is then if, due to the lack of synchronization among RFID readers, RRE can leave uncovered tags. We define the following safety property which should hold for any distributed algorithm for the redundant-reader elimination problem and prove that RRE satisfies it.

Safety: An algorithm for the redundant-reader elimination problem is said to be safe, if it will not turn off RFID readers that cover RFID tags not covered by active readers.

Claim: RRE is safe.

Proof: Let us assume that a tag $\mathtt{T}_1$ is situated inside the interrogation zones of two readers, $\mathtt{R}_1$ and $\mathtt{R}_2$. Furthermore, $\mathtt{R}_1$ covers fewer tags than $\mathtt{R}_2$. Then, it is likely for $\mathtt{R}_1$ to start the second step of RRE before $\mathtt{R}_2$ has succeeded writing its tag count on its covered tags. Then, both $\mathtt{R}_1$ and $\mathtt{R}_2$ will believe to be the locker of $\mathtt{T}_1$. However, $\mathtt{T}_1$ will not be left uncovered, since both $\mathtt{R}_1$ and $\mathtt{R}_2$ are required to stay active. This will only decrease the number of redundant readers able to be simultaneously deactivated. ■

Complexity of RRE: $\mathtt{T}_{\mathtt{RRE}} = \mathtt{O}(\gamma \log \beta \log \psi)$.

Proof: The complexity of RCA, is $\mathtt{O}(\gamma \log \beta \log \psi)$ (see Section III-A). The first step of RRE, where each RFID reader sends a write command to all its tags, takes $e \log \psi$ epochs. The second step, where RFID readers send queries to each of their tags, takes $\gamma \mathtt{e} \log \psi$ epochs. Thus, $\mathtt{T}_{\mathtt{RRE}} = \mathtt{O}(\gamma \log \beta \log \psi)$. ■

VI. Simulation Results

All of our experiments are performed by randomly (uniformly) deploying RFID tags and readers in a $1000 \times 1000m^2$ domain. We first evaluate the accuracy of RCA in allowing readers to detect the tags situated in their interrogation zones and then analyze the efficiency of RRE in terms of the number of redundant readers detected.

A. RCA

In this section we experimentally analyze the accuracy and message overhead introduced by our randomized solution to the reader collision problem. We compare the performance of RCA with the simple tree walk algorithm [11](TWA) and with a version of RCA, that we call RCAv.1. In RCAv.1, a reader sends each query the maximum number of times, irrespective of the result of the query. Note that in RCA, a reader will not repeat a query if the result is a success, that is, it receives an answer from a tag or it detects a tag collision. All our experiments are performed by randomly (uniformly) deploying tags and readers in a 1000 units $\times$ 1000 units square.

In the first experiment we randomly place $\gamma = 4000$ tags and $\psi = 500$ readers having an interrogation radius of 50 units in the 1000 units $\times$ 1000 units square area. Moreover, we increase the number of repetitions (epochs) per query from 1 to $2 \log \psi$. Fig. 4(a) shows the average number of tags detected by a reader, compared with the average number of tags actually placed in the interrogation zone of the reader. When RCA is used, the number of tags discovered by a reader quickly converges to the number of tags placed in its interrogation zone. For 9 ($\log \psi$) repetitions per query RCA allows any reader to discover *all* the tags placed in its interrogation zone. This shows that the result of Theorem 1 (see Section **??**) is valid in a worst case scenario. Moreover, for a realistic distribution of readers and tags, the constant hidden in the big-oh notation is small, 1. Fig. 5(b) shows the corresponding number of messages per query generated by RCA in this scenario. As expected, the growth in the number of messages sent per query is linear with the total number of query repetitions. However, when each query is repeated up to 9 ($\log \psi$) times, RCA generates on average only half, 4.5 messages. Thus, in the following experiments we consistently repeat each query of RCA at most $\log \psi$ times.

Next, in a configuration of $\gamma = 4000$ tags and $\psi = 500$ readers, each with an interrogation radius of 50 units, we increase the number of time frames per epoch employed by RCA from 1 to 38. Fig. 5(a) shows our observations. As expected, the detection accuracy of readers improves for more time frames per epoch, quickly converging to 100%. For example, for 9 ($\log \psi$) frames per epoch, our algorithm enables readers to detect on average 99.3% of the tags placed in their interrogation zone. However, *all* the tags are detected when the number of frames per epoch is 18 ($2 \log \psi$). This is due to the uniform distribution of tags and readers. When b balls are thrown uniformly at random in b bins, the maximum number of balls in any bin is $O(\log b)$ [13].

For the same scenario, Fig. 5(b) shows the average number of times a query needs to be repeated by a reader, in order to read all the tags situated in its interrogation zone. For 18 ($2 \log \psi$) time frames per epoch, the required number of repetitions per query is around 4.5, half of the maximum number of query repetitions. Thus, by not repeating successful queries, our randomized algorithm saves on average more than

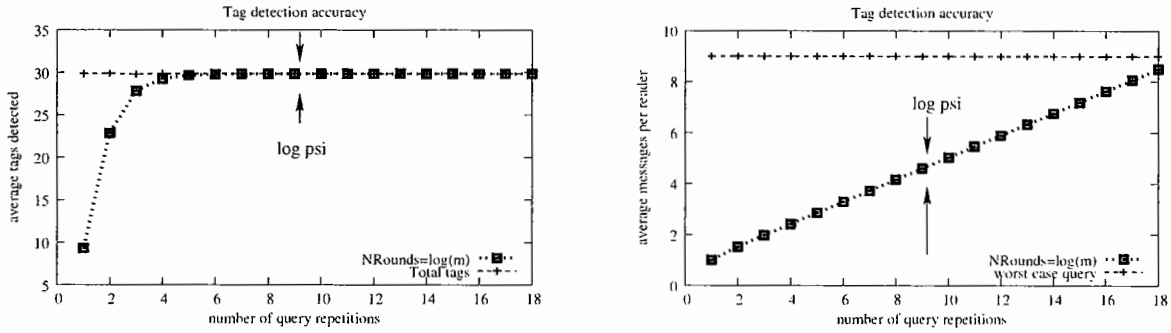


Fig. 4. Performance of RCA when the number of epochs per query grows from 1 to $2\log\psi$, for a total of 500 readers and 4000 tags. The number of time frames per epoch is $2\log\psi$, where ψ is the number of readers.

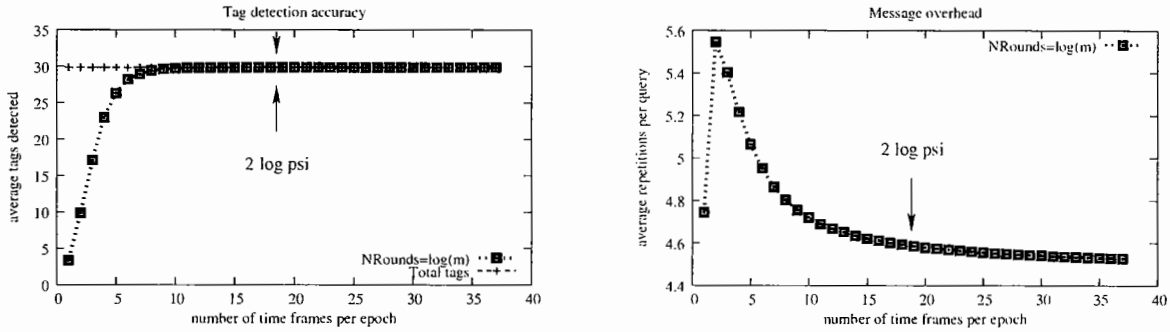


Fig. 5. Performance of RCA when the number of time frames per epoch increases from 1 to 38, for a total of 500 readers and 4000 tags. The number of time epochs per query is $\log\psi$, where ψ is the number of readers.

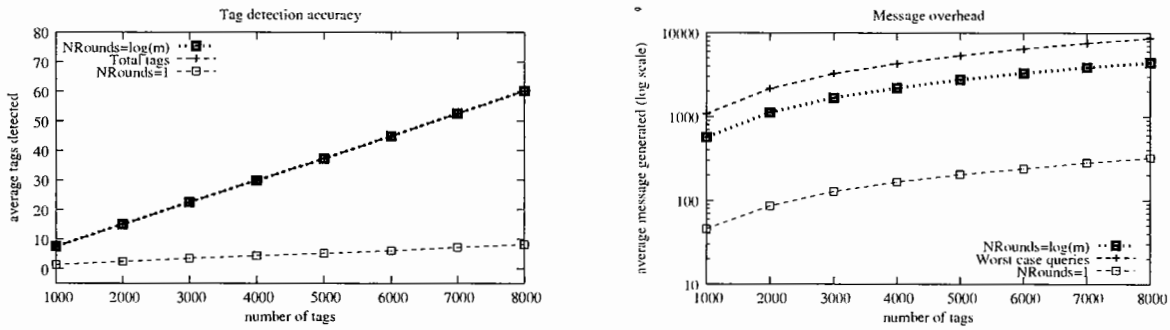
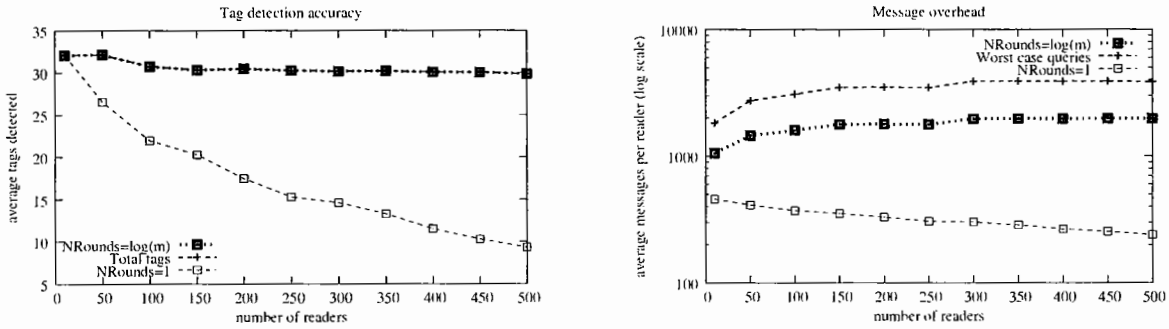


Fig. 6. Scalability of RCA and TWA when the number of tags increases from 1000 to 8000, while the number of readers is 1000.

4 messages per query. Moreover, this shows that for realistic scenarios the necessary number of time frames per query can be significantly smaller than ψ, the value used in Theorem 1 (see Section ??). As a consequence, all the following simulations evaluate RCA using only $2\log\psi$ time frames per epoch.

In the following experiment, we randomly place 1000 readers with an interrogation radius of 50 units and increase the number of randomly placed tags from 1000 to 8000. As mentioned above, the number of time frames per epoch is set to $2\log\psi$ both for RCA and TWA. Fig. 6(a) shows the accuracy of readers, as the average number of tags detected, when using RCA with $\log\psi$ repetitions per query and TWA with one message per query. While RCA is very accurate, 99.94%, TWA has 14% accuracy. Fig. 6(b) shows the corresponding number of messages per readers generated by the two algorithms, on a logarithmic scale, compared with the total number



(a) (b)

Fig. 7. Scalability of RCA and TWA when the number of readers increases from 10 to 500, for 4000 tags.

of messages per reader generated by RCAv.1. While RCA generates 10 times more messages than TWA, this is simply due to the fact that TWA performs 3 times less query types than RCA. This is also the reason behind TWA's inaccuracy. However, RCA halves the number of messages generated by RCAv.1, by not repeating answered queries.

In order to evaluate the reader scalability of RCA and TWA, we place 4000 tags and increase the number of readers randomly deployed from 10 to 500. The interrogation radius of readers is set to 50 units. Fig. 7(a) shows the accuracy of RCA and TWA. For a small number of readers, TWA accurately detects the tags deployed in the interrogation zone of readers. This is because the interrogation zones of readers barely intersect, practically eliminating reader collisions. However, as the number of readers increases, effectively increasing the overlapping areas of the interrogation zones of readers, the accuracy of TWA constantly decreases. In contrast, RCA, by using $\log \psi$ epochs per query is accurate, consistently discovering *all* the tags deployed. This accuracy comes at the expense of more messages. Fig. 7(b) shows the corresponding average number of messages generated by RCA, TWA and RCAv.1. The values are shown on a logarithmic scale. As observed in the previous experiment, TWA is message efficient, since few queries are successful, leaving unexplored most of the name space of tags. However, the number of messages generated by RCA quickly saturates and is only half of the messages generated by RCAv.1.

The last experiment evaluates the performance of RCA when the interrogation radius of readers increases from 40 units to 100 units, while the number of readers randomly deployed is 500 and the number of tags is 4000. Fig. 8(a) shows the accuracy of RCA compared with TWA. RCA discovers *all* the tags until the interrogation radius reaches 85m. However, even for an interrogation radius of 100 units, RCA has a 94% accuracy. As the interrogation radius increases so does the size and number of intersections of interrogation zones of readers. Since the number of epochs per query, $\log \psi$ and the number of time frames per epoch is constant, more collisions are generated, leading to a decreased accuracy. However, the performance of TWA is considerably inferior. When the interrogation radius of readers is 100 units, readers running TWA discover only 5% of the tags detected by readers running RCA. Fig. 8(b) shows the corresponding number of messages generated by the same algorithms. It confirms the results shown in Fig. 6(b) and Fig. 7(b). TWA generates only a fraction of the messages generated by RCA, since the number of queries correctly detected as successful is very small. However, we consistently reduce the number of messages sent by eliminating repetitions of successful queries.

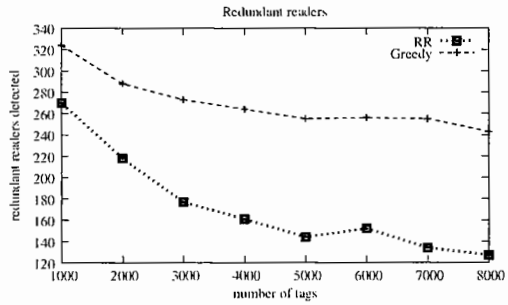
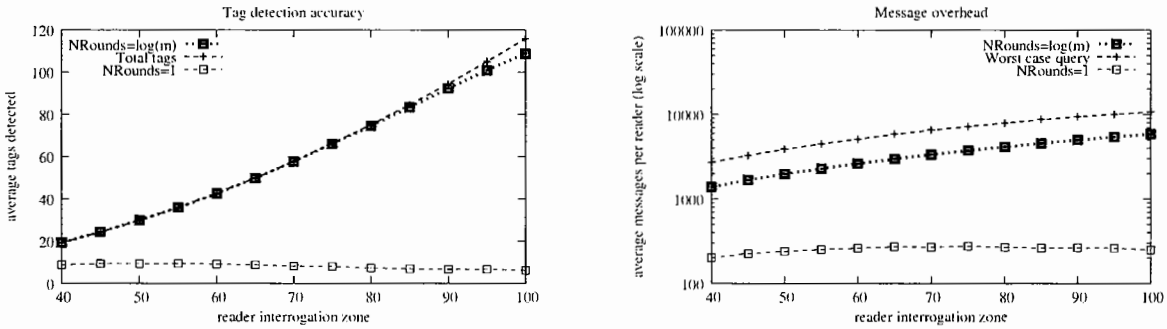


Fig. 9. Number of redundant readers discovered by RRE and Greedy when the number of RFID tags randomly deployed increases from 1000 to 8000. The number of RFID readers is constant, 500, throughout this experiment.

B. *Efficiency of RRE*

We now compare the performance of our redundant-reader detection algorithm with a centralized greedy approximation algorithm for the redundant-reader problem. The comparison is done in terms of the number of RFID readers able to be turned off simultaneously.



(a) (b)

Fig. 8. Performance of RCA and TWA when the interrogation zone of the readers increases from 40 units to 100 units, for 500 readers and 4000 tags.

The centralized greedy algorithm, GREEDY, sequentially selects the unvisited RFID reader with the highest density of covered, unvisited RFID tags. It then marks the selected RFID reader and its covered RFID tags as visited. GREEDY stops when there are no more unvisited tags. The set of visited RFID readers remain active and the others can be safely deactivated. GREEDY is safe, in the sense that deactivated RFID readers will not leave tags uncovered (see Section V-B). The GREEDY algorithm is however difficult to implement, since it requires centralized knowledge of the reader network.

In the first experiment we randomly place 500 RFID readers and between 1000 and 8000 RFID tags in the 1000×1000 m^2 domain. Figure 9 shows the number of redundant RFID readers discovered by RRE and GREEDY. For fewer RFID tags deployed, RRE is reasonably close to GREEDY, by discovering 83% of the redundant readers discovered by GREEDY. As the number of RFID tags increases, the performance of RRE relative to GREEDY degrades, but it always discovers over 50% of the redundant readers of GREEDY. Both GREEDY and RRE discover less redundant readers as the number of deployed RFID tags increases. Both algorithms base their decision on the number of RFID tags covered by readers. By increasing the RFID tag

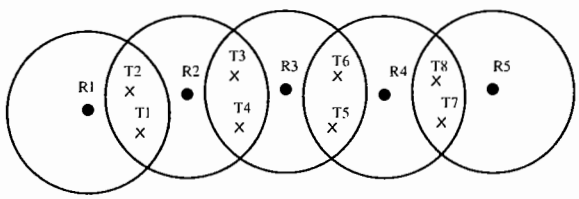


Fig. 10. Difficulty of consistently breaking ties. The optimal solution keeps only R_2 and R_4 active. However, in a scenario where R_2, R_3 and R_4, each covering 4 tags, lock a different set of tags, all of them will have to be active.

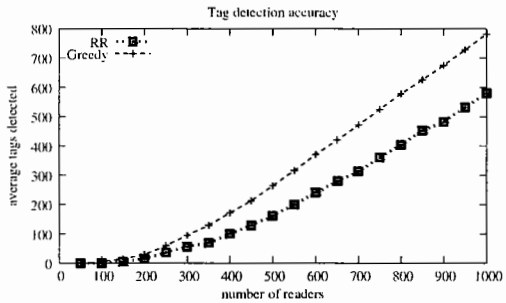


Fig. 11. Number of redundant readers discovered by RRE and Greedy when the number of RFID readers randomly deployed increases from 50 to 1000, for a total of 4000 RFID tags.

density, the distribution of RFID tags per reader becomes more uniform, making it more difficult to choose good, active RFID readers. However, the decrease is more acute for RRE, since in scenarios where readers whose interrogation zones overlap cover equal numbers of tags, consistently breaking ties becomes a difficult problem. We illustrate such a scenario in Figure 10, where each of readers R_2, R_3 and R_4 covers four tags. While the

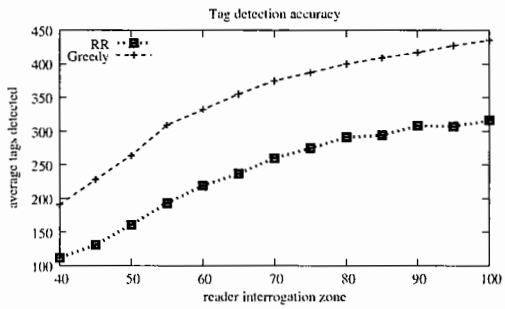


Fig. 12. Number of redundant readers discovered by RRE and Greedy when the interrogation radius of RFID readers increases from 40 to 100m. The number of RFID readers is 500 and the number of RFID tags is 4000, for the entire duration of the experiment.

optimal solution requires only R_2 and R_4 to be active, we can imagine a run of RRE where R_4 locks $T_5, .., T_7$, R_3 locks T_3 and T_4 and R_2 locks T_1 and T_2, effectively requiring all three readers to be active. The example can be easily extended, and one can see that in the worst case RRE can require $2r - 1$ active readers, where r would be sufficient. This degenerate worst case is, however, rare. Moreover, as noted before, the performance of GREEDY comes with the high cost of collecting all reader network information at a central point.

The second experiment compares the performance of RRE and GREEDY when the number of randomly deployed RFID readers increases from 50 to 1000, when the total number of RFID tags is 4000. Figure 11 shows the results of this experiment. For scarce deployment of RFID readers, very few of the readers are redundant. As their density increases, however, so does the number of redundant readers. For example, for 1000 RFID readers, GREEDY discovers almost 800 to be redundant. While initially RRE is very accurate, as the number of RFID readers increases, RRE discovers fewer redundant readers. However when between 500 to 1000 readers are deployed, RRE consistently discovers more than 80% of the redundant readers of GREEDY. The difference is again due to the difficulty in breaking ties in RRE. As the number of deployed RFID readers increases, the number of readers whose interrogation zones overlap, also increases, generating more contentions.

The final experiment measures the dependency between the number of redundant readers discovered by RRE and GREEDY and the interrogation zones of RFID readers. We randomly deploy 500 RFID readers and 4000 RFID tags, and increase the interrogation radius of readers from 40 to 100m. Figure 12 shows that as expected, with the increase in the interrogation radius of RFID readers, both RRE and GREEDY discover an increasing number of redundant readers. This is because active readers cover larger areas, effectively necessitating fewer active readers to cover all the tags. Note that while RRE discovers fewer redundant readers than GREEDY, the difference is almost constant for smaller interrogation zones. Due to an increase in the number of interrogation zone overlaps, leading to an increased difficulty of breaking ties, the difference between GREEDY and RRE increases slightly for large interrogation zones.

VII. Related Work

The reader-collision problem in RFID systems was first documented in [15]. The solution proposed, of allocating different frequencies to interfering RFID readers, is centralized. A simple decentralized version, where readers listen for collisions and use randomized backoff when detecting one, is discussed. In contrast, our work assigns different time slots for transmitting RFID readers. Moreover, our solution guarantees w.h.p. that each RFID reader is able to correctly read all the RFID tags placed in its interrogation zone.

Perhaps closest to our goal of correctly reading covered RFID tags is the work of Waldrop et. al [16]. They propose Colorwave, a decentralized MAC protocol for RFID reader networks whose purpose is to allocate disjoint time slots for reader transmissions. The protocol is based on the presence of an interference graph whose links denote interference between the end-points corresponding to RFID readers. Hence, an interesting extension to this work would be a description of the interference graph construction. As shown in Figure **??**, interference at certain RFID tags is difficult to detect, since even the presence of such tags may not be known.

The problem of coverage of a set of entities has been studied in a variety of contexts. In the area of wireless sensor networks, Tian and Georganas [7] present an algorithm for detecting sensors whose coverage area is completely covered by other sensors. A sensor turns itself off only when each sector of its coverage disk is covered by another sensor. Zhang and Hou [8] provide a distributed algorithm for extending the network lifetime by turning off "redundant" sensors. Their mechanism for deciding a sensor to be redundant requires a sensor to divide its coverage area into small grids and then using a bitmap to indicate whether the center of each square of the grid is covered by some other sensor. Ye et al. [9] present an algorithm that extends the network lifetime by maintaining a necessary set of working sensors and turning off redundant ones. A sensor is alternatively sleeping or active. When a sensor wakes up, if it has an active sensor inside its transmission range, it turns off again. Slijepcevic and Potkonjak [17] introduce a centralized algorithm for finding the maximum number of disjoint subsets of sensors, where each subset completely covers the same area as the entire set of sensors. All the above work uses a definition of coverage in terms of continuous areas. Our goal is however to detect a discrete set of points in the coverage area of a RFID reader network. Moreover, we define coverage only in terms of the set of discrete points, RFID tags. While this approach has the potential to discover more redundant RFID readers, the problem is complicated by the scarce resources of RFID tags.

Medium Access Control protocols for wired and wireless networks share several details with our reader collision avoidance algorithm. The first MAC protocol, proposed for packet radio networks, is ALOHA [18]. When the transmission of a node results in collision, the node

must wait for a random interval before retransmitting. However, RFID systems do not have the mechanisms to detect collisions occurring at tags, making ALOHA unsuitable for avoiding reader collisions. Multiple access with collision avoidance (MACA) [19] is a protocol that employs a handshake to avoid hidden-node problems. The sender broadcasts an RTS message and the receiver replies with a CTS message. All the nodes that hear the RTS and CTS messages delay their transmissions. Such a protocol cannot be used in RFID system, since the purpose of an RFID reader is to detect *all* the RFID tags in its interrogation zone. Such a reader does not know the identities of the RFID tags and thus cannot send individual RTS messages. Moreover, the simultaneous reception of CTS messages initiated by RFID tags leads to tag collision problems. Carrier sensing multiple access with collision detection (CSMA/CD) [20], employed in the standard Ethernet is based on the ability of nodes to detect collisions. Upon detecting a collision, a node waits for a random interval before retransmitting. In case of subsequent collisions, the node waits twice as long before attempting to retransmit, also known as exponential back-off. However, as noted before RFID systems lack the ability of detecting remote collisions.

VIII. Conclusions

In this paper we address two important problems in wireless RFID systems. The first problem, of accurately detecting the RFID tags covered by each RFID reader, is made difficult by reader collisions occurring at remote RFID tags. The second problem relates to extending the lifetime of the reader network by detecting and temporarily turning off redundant readers. We define redundancy in terms of discrete sets of points, RFID tags, and prove that the optimization version of the problem is NP-complete. For both problems, we present distributed and localized algorithms, based on a randomized querying technique, that ensures, w.h.p., the accurate receipt of reader queries by RFID tags. We provide a probabilistic analysis of the algorithms. Our extensive simulations show that our reader collision avoidance algorithm is very accurate and our redundant-reader elimination heuristic is efficient, when compared to a centralized greedy approximation of the optimum solution.

References

[1] S.E. Sarma. Towards the five-cent tag. Technical Report MIT-AUTOID-WH-006, MIT Auto ID Center, 2001.

[2] Edwin Kalischnig. RFID: Making sense of sensor-based technology. Manufacturing and Logistics IT, July 2004.

[3] Ann Bednarz. Wireless technology reshapes retailers. Network World, 12 August 2002.

[4] SkyeTek. http://www.skyetek.com/readers_Mini.html, January 2004.

[5] SmartCode. http://www.smartcodecorp.com/.

[6] Battery Assisted Passive Microwave RFID tags. http://www.alientechnology.com/products/rfid-battery/bap_tags.php.

[7] Di Tian and Nicolas D. Georganas. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *Proceedings of the 1st ACM WSNA*, pages 32–41. ACM Press, 2002.

[8] Honghai Zhang and Jennifer Hou. Maintaining coverage and connectivity in large sensor networks. In *International Workshop on Theoretical and Algorithmic Aspects of Sensor, Ad hoc Wireless and Peer-to-Peer Networks*, Feb 2004.

[9] Fan Ye, Gary Zhong, Songwu Lu, and Lixia Zhang. Peas:a robust energy conserving protocol for long-lived sensor networks. In *23rd IEEE ICDCS*, 2003.

[10] B. Carbunar, A. Grama, J. Vitek, and O. Carbunar. Coverage-preserving redundancy elimination in sensor networks. In *IEEE SECON*, 2004.

[11] Sanjay E. Sarma, Stephen A. Weis, and Daniel W. Engels. RFID systems and security and privacy implications. In *CHES '02*, pages 454–469. Springer-Verlag, 2003.

[12] Henri E. Bal, Raoul Bhoedjang, Rutger Hofman, Ceriel Jacobs, Koen Langendoen, Tim Ruhl, and M. Frans Kaashoek. Performance evaluation of the Orca shared-object system. *ACM Trans. Comput. Syst.*, 16(1):1–40, 1998.

[13] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[14] R. Fowler, M. Paterson, and S. Tanimoto. Optimal packing and covering in the plane are np complete. *Information Processing Letters*, 12(3):133–137, 1981.

[15] D. W. Engels and S. E. Sarma. The reader collision problem. In *IEEE International Conference on Systems, Man and Cybernetics*, 2002.

[16] J. Waldrop, D.W. Engels, and S.E. Sarma. Colorwave: a MAC for RFID reader networks. In *Wireless Communications and Networking (WCNC)*, 2003.

[17] Sasa Slijepcevic and Miodrag Potkonjak. Power efficient organization of wireless sensor networks. In *IEEE ICC*, 2001.

[18] N. Abramson. The ALOHA system: Another alternative for computer communications. In *AFIPS Conf. Proc., Fall Joint Computer Conf*, 1970.

[19] P. Karn. MACA a new channel access method for packet radio. In *ARRL/CRRL Amatuer Radio 9th Computer Networking Conf.*, 1990.

[20] Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications. Institute of Electrical and Electronics Engineers, 1996.