J. Parallel Distrib. Comput. 69 (2009) 180–196

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

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

Efficient tag detection in RFID systems

Bogdan Carbunar [a,*], Murali Krishna Ramanathan [b], Mehmet Koyutürk [c], Suresh Jagannathan [b], Ananth Grama [b]

[a] Motorola Labs, 1295 E. Algonquin Road. IL05 2nd Floor, Schaumburg, IL 60195, United States

[b] Department of Computer Science, Purdue University, West Lafayette, IN 47907, United States

[c] Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, OH 44106, United States

ARTICLE INFO

Article history:
Received 14 January 2008
Received in revised form
19 May 2008
Accepted 22 June 2008
Available online 8 July 2008

Keywords:
RFID systems
Tag coverage
Reader redundancy
Reader collision avoidance

ABSTRACT

Recent technological advances have motivated large-scale deployment of RFID systems. However, a number of critical design issues relating to efficient detection of tags remain unresolved. In this paper, we address three important problems associated with tag detection in RFID systems: (i) accurately detecting RFID tags in the presence of reader interference (*reader collision avoidance problem*); (ii) eliminating redundant tag reports by multiple readers (*optimal tag reporting problem*); and (iii) minimizing redundant reports from multiple readers by identifying a minimal set of readers that cover all tags present in the system (*optimal tag coverage problem*). The underlying difficulties associated with these problems arise 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. In this paper we present a randomized, distributed and localized Reader Collision Avoidance (RCA) algorithm and provide detailed probabilistic analysis to establish the accuracy and the efficiency of this algorithm. Then, we prove that the optimal tag coverage problem is NP-hard even with global knowledge of reader and tag locations. We develop a distributed and localized Redundant Reader Elimination (RRE) algorithm, that efficiently identifies redundant readers and avoids redundant reporting by multiple readers. In addition to rigorous analysis of performance and accuracy, we provide results from elaborate simulations for a wide range of system parameters, demonstrating the correctness and efficiency of the proposed algorithms under various scenarios.

1. Introduction

Radio Frequency Identifier (RFID) is an increasingly deployed technology for tagging and uniquely identifying objects. Unlike barcodes, RFID enables simultaneous detection of multiple, distant, and non-line-of-sight objects. There are hundreds of millions of RFID tags currently deployed; it is believed that this number will reach tens of billions within seven years [20]. The use of RFID systems is primarily motivated by their net savings, low deployment cost, and accuracy. As an example, Wal-Mart anticipates savings of billions of dollars from tagging pallets and individual products, while the International Air Transport Association (IATA) projects annual industry savings of $760 Million from RFID luggage tags [16]. Apparel vendor Gap documented an increase of accuracy from 85% to 99.9% when using RFID technology [5].

RFID systems have two types of components, RFID transponders (tags), placed on objects and RFID transceivers (tag readers). RFID tags store information using a small integrated circuit and communicate using an antenna. RFID readers are capable of reading the information stored on tags placed in their vicinity and communicate it to a host computer. Most tags are passive, *i.e.* they do not require battery power. Instead, passive tags use the energy of the reader's signal to fetch, process, and communicate stored data.

The past few years have witnessed tremendous advances in RFID technology beyond reduction in size and price of components. Notably, diverse solutions for wireless readers are now supported, including quarter-sized wireless readers [28] compatible with Crossbow [10] sensors, Bluetooth enabled readers [4], SD card reader/writers [32], and reader enabled phones [22]. These developments broaden the scope and utility of RFID systems, enabling their on-line deployment. On the other hand, convenient, remote ad hoc deployment of wireless readers poses several problems.

Reader collision avoidance: One problem, *reader collision* [12], occurs when co-located readers are simultaneously active. Specifically, reader collisions occur at tags situated in the vicinity of two or

* Corresponding author.
E-mail addresses: carbunar@motorola.com (B. Carbunar), rmk@cs.purdue.edu (M.K. Ramanathan), koyuturk@eecs.case.edu (M. Koyutürk), suresh@cs.purdue.edu (S. Jagannathan), ayg@cs.purdue.edu (A. Grama).

0743-7315/$ – see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2008.06.013

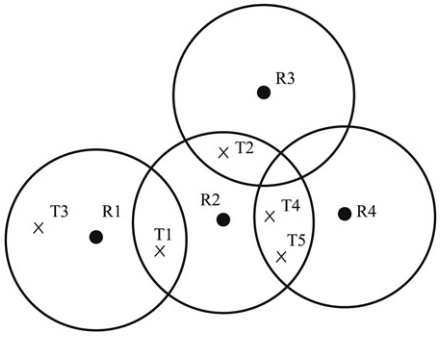

Fig. 1. Example reader and tag deployment. Readers are denoted by small dark circles and their RF interrogation zones by larger disks. Tags are denoted by small crosses. In this example, all the readers except R_1 are redundant, however, both R_1 and R_2 need to report their covered tags. Moreover, tag T_1 is redundantly covered by R_1 and R_2 and may be reported twice. RRE assigns tag T_1 to reader R_2, which will be the only reader reporting it.

more readers that simultaneously interrogate tags. Such tags may be unable to correctly decode reader queries, leading to undesirable behavior. Consider, for example, the reader and tag deployment in Fig. 1. In the figure, readers are represented by dots, tags are represented by crosses, and the interrogation zone of a reader is represented by circles centered around readers. If readers R_1 and R_2 interrogate tags at the same time, neither of them may detect tag T_1. This is because T_1 resides at the intersection of interrogation zones of the two readers, and consequently responds to both of their queries. This causes existing tag detection protocols to potentially report incorrect data, or to not terminate at all.

Current protocols for avoiding reader collisions are based either on time scheduling of reader transmissions, spatial isolation of readers or frequency assignment to readers, or a combination of these techniques. Solutions may either require a centralized entity or are completely decentralized.

Time division multiple access (TDMA) techniques are used when networked readers may be instructed to read at different times. The absence of global topology and reader and tag location information requires a distributed implementation of TDMA. An instance of a distributed TDMA solution is the Colorwave protocol of Waldrop et al. [31]. Colorwave attempts to assign a tag interrogation time slot to each reader in the network, such that no two readers whose transmissions interfere are assigned the same slot. To achieve this, Colorwave assumes that each reader knows the set of readers with whom it interferes, and can communicate with them. Note that Colorwave is independent of tag locations. For a given reader network instance, the Colorwave solution works for any tag deployment. However, building the interference set is a challenging problem in itself, as the interference range of wireless devices may well exceed their transmission range. In the worst case a reader may need to contact $O(n)$ other readers, where n is the reader network's size, in order to build its interference set. Moreover, the assumption of reader communication capabilities, restricts the applicability of the protocol to reader devices that can communicate. Even if readers have additional wireless interfaces (e.g., 802.11x, 802.15.4 or Bluetooth) their power requirements may significantly reduce the network's lifetime.[1]

Spatial isolation relies on predefined reader deployment, with antennae designed to read clearly delineated areas, coupled with the use of RF shielding materials to prevent other readers from detecting tags in the same area. The work of Zhou et al. [36], based on the Spatial Time Division Multiple Access (STDMA) protocol is an instance of a hybrid between spatial, temporal and frequency division mechanisms. In the case where the tag distribution is unknown, the authors attempt to solve the problem assigning readers to frequency channels in each time slot, such that each location of the deployment space is well-covered by some reader in one of the time slots. Then the authors extend and optimize their solution for the case where the tag distribution is known. This approach makes two important assumptions, which restrict its applicability. First, a planned deployment of readers is assumed, allowing deployers to perform RF site surveys to measure the location and interference patterns of readers. However, spatial isolation is difficult to implement in un-orchestrated, on-line reader deployment scenarios. Second, the existence of a central entity processing this information and providing a channel and slot assignment to each reader is assumed.

The frequency assignment solution for the reader collision problem consists of allocating different frequency channels to interfering readers, that can then be scheduled for reading tags simultaneously. Deolalikar et al. [11] proposed a graph theoretic approach for this problem. They made the observation that if the interference graph for a reader network can be converted into a bipartite graph, the frequency assignment is to simply assign one frequency to readers in one of the partitions and another frequency to the readers in the other partition. The authors show that the conversion of the interference graph can be achieved in a succession of steps, where each step can be either removing one edge or one vertex from the graph. They propose heuristics for choosing the best edges and vertices for removal, both based on a "correlation" metric. The correlation of two readers is defined to be the number of tags covered by both readers. This approach suffers thus from two problems, (i) it requires the existence of an interference graph and (ii) it requires the readers to read the tags, before being able to assign frequencies to readers, for reading the tags.

In this paper, we consider the *reader collision avoidance problem* in a more general setting, *i.e.*, assuming arbitrary tag distribution and reader deployment, and no communication among readers. We develop RCA, a randomized, distributed, and localized time division algorithm. RCA avoids reader collisions *with high probability*, allowing readers to accurately detect tags in their vicinity. RCA does not require inter-reader communication capabilities, rather, it relies on existing reader-to-tag communication capability. We prove that when the interrogation zones of ψ readers overlap, RCA requires at most $O(\log\ \psi)$ reader-to-tag retransmissions. We further show through extensive simulations that for realistic reader and tag deployments RCA generates very few retransmissions, while consistently discovering over 99.9% of all deployed tags.

Optimal tag reporting: Motivated by considerations of power at readers, as well as minimizing the overheads of multiple reader resolution, we consider the problem of minimizing redundant tag reports—the *optimal tag reporting problem*. Consider the scenario illustrated in Fig. 1. Here, if readers report all tags in their interrogation zone without any optimization, tags T_1, T_2, T_4, and T_5 are reported twice, generating nine tag reports for five tags. While duplicate tag reports will be eventually eliminated by the host system, the number and duration of wireless transmissions may be reduced by detecting and eliminating redundancy as early as possible, *i.e.*before any communication between the reader and the host system occurs.

Optimal tag coverage: We further introduce and study the related *optimal tag coverage problem*. This problem follows from the observation that a reader that covers only tags covered by other

[1] We have tested the 802.11b cards of Motorola A910 phones and found that even in idle mode they reduce the battery lifetime by more than 45%.

readers as well, is *redundant*.[2] Consequently, the number of readers that need to use their wireless interface can be minimized by identifying a minimal subset of readers in which no reader is redundant (with respect to the subset). This is the optimal tag coverage problem, as illustrated in Fig. 1. Observe that all readers except R_1 in this deployment are redundant. On the other hand, the minimum number of readers that need to report detected tags to ensure complete tag coverage is two, since no reader is redundant in the set $\{R_1, R_2\}$.

We first prove that even with centralized knowledge of tag distribution and reader deployment, the problem of finding the optimal number of readers that need to report tags is NP-hard. We then propose RRE, a decentralized and localized algorithm for the optimal tag coverage problem. Similar to RCA, RRE requires only basic reader-to-tag communication capabilities. Furthermore, we show that a side effect of RRE is to allow each tag to be reported by a single reader, effectively eliminating redundant tag reports, *i.e.*providing a solution to the optimal tag reporting problem.

Our simulation results show that the performance of RRE is close to that of a centralized greedy heuristic for the minimum set cover problem. We also show that, by ensuring that each tag is reported only once, RRE generates a significantly lower number of (redundant) tag reports. Specifically, RRE generates between 20% and 600% fewer tag reports than a protocol that does not eliminate redundancy.

This paper is organized as follows: we present background material, the RFID system model, and the problems addressed in this work in Section 2. In Section 3, we discuss related work. Section 4 describes a randomized querying technique for avoiding reader collisions, used in both RCA and RRE. The technique is described in the context of RCA. We prove the NP-hardness of the optimal tag coverage problem in Section 5 and present the details of our solution, RRE, in Section 6. Simulation results of both algorithms are presented in Section 7. We conclude our discussion in Section 8.

2. Preliminaries

We initiate our discussion with an overview of RFID systems and the capabilities of existing tags and readers. In Section 2.2, we briefly describe the tree walking algorithm for resolving tag collisions. In Section 2.3, we formally define the problems addressed in this paper. Finally, in Section 2.4, we describe the system model considered.

2.1. Overview of RFID systems

RFID tags contain limited memory resources for storing unique identification, and information relating to the objects with which they are associated (attached). They also contain an antenna, used for communication with readers. There are two kinds of tags: passive and active. Passive tags do not need batteries, but are powered by RF energy from the reader. In addition to the memory chip and antenna, active tags are equipped also with a battery and a transmitter, allowing them to initiate transmissions.

While active tags have a greater communication range and can operate autonomously, they are more expensive. The price efficiency of tags is the dominant factor determining their wide deployment. Most current RFID applications use only passive tags, thus, this paper focuses on passive tags. For most tags, the chip and antenna are mounted on a base and encapsulated with thermoplastic. Their length ranges from 1/16 inch to more than 6 inches. Tags can be paper thin or embedded into materials allowing them to withstand high temperatures and chemical environments. They can be flexible enough to be embedded within an adhesive label and run through a printer.

Most RFID systems use either the low frequency LF (125–134.2 kHz and 140–148.5 kHz), high frequency HF (13.56 MHz) or ultra high frequency domains UHF (860–960 MHz) for transmission. HF tags typically cost less and are better suited for tagging water or liquid-bearing objects because the longer wavelengths of HF systems are less susceptible to absorption. A UHF tag can be made to work in these conditions, but its effective read range is dramatically reduced. The ISO/IEC 15693 standard has enabled the global acceptance of 13.56 MHz RFID technology. Examples of HF tags include TI's Tag-It HF-1, Philips' I-Code SLI, Infineon's my-d SRF55VxxP or ST Microelectronics' LRI512. These tags have factory programmed 64 bit identifiers and between 112 and 1000 bytes of memory available for read/write operations.

Signals from RFID readers activate compatible tags within their interrogation zone. The interrogation zone is defined to be the area around a reader where tags can receive the reader's signal, process it and send back a response that can be correctly decoded by the reader. The information decoded by the reader is passed to host computing systems where it is further processed, according to the application. In addition to locating, activating, and receiving transmissions from RFID tags, RFID reader-writers also have the ability to send data back to read/write-capable tags in order to append or replace data.

Readers may themselves be fixed or portable. Fixed readers are usually attached to antennae that are designed to detect the tags within a specified area. These units typically collect data from products traveling through loading dock doors, conveyor belts, gates and doorways, or even cars on tollways. Portable, wireless readers can be moved to detect remote tags, in areas where wiring or antenna placement could be difficult. Readers with various wireless communication capabilities exist in today's market. SkyeTech's SkyeRead M1 [28] reader is compatible with Mica Motes [10] and IDBlue [4] is a handheld Bluetooth 13.56 MHz RFID reader, compatible with devices ranging from PDAs to PCs. Major cellular phone manufacturers provide phones with embedded RFID readers [22].

The size, portability, and price of such wireless readers motivates their deployment and organization into ad hoc networks. The network can not only detect tags but also relay tag reports of remote readers toward the host system. Several applications benefit from arbitrary deployments of reader networks, including supply chain management, warehouse management [18], baggage management in airports [2,16], and tracking participants at marathons [26]. In an arbitrary deployment the placement of readers does not follow pre-defined constraints so no assumptions about the location or interference patterns of readers can be made.

Reader networks are currently supported by a number of vendors. PGS Electronics [25] provides a complete solution for wireless RFID integration and Reva Systems [24] recently proposed a protocol called Simple Lightweight RFID Reader Protocol (SLRRP). SLRRP defines how readers convey configuration, control, status, and tag information between RFID reader network device managers and other readers in an IP-based network, either wired or wireless.

We consider two models of wireless reader network deployments, each with a different set of networking capabilities. While the results presented in this paper are not restricted to these models, we believe them to be of independent value.

The first model considers only the deployment of RFID readers that are equipped with cellular interfaces (e.g., RFID enabled

[2] In multi-hop wireless reader networks, redundant readers may still need to use their wireless interface to forward tags detected by other readers. In this case, however, eliminating duplicate tag reports is even more important, since redundancy may generate a large number of transmissions.

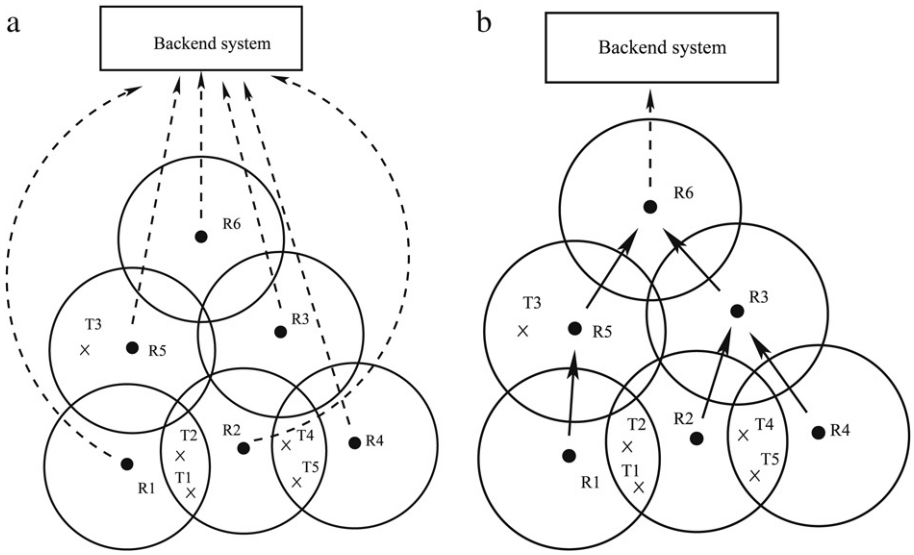

Fig. 2. Example of redundant tag reporting. Readers are denoted by small filled circles and their RF interrogation zones by larger disks. Dotted arrows represent cellular links and plain arrows represent reader-to-reader wireless links. Small rectangles represent tags. (a) Wireless readers may only communicate with the host system. If readers report tags directly to the host server, readers R_1 and R_4 can safely *not* transmit. If they do, the duplicate tag reports are eliminated by the host system. (b) The same reader and tag placement but wireless readers may now communicate with each other and with the host system. The root node R_6 needs to communicate with the host system. All other readers route their reported tags toward R_6. If R_1, R_2 and R_4 all communicate their detected tags, tags T_1, T_2, T_4 and T_5 are reported more than once. While the duplicate reports of T_4 and T_5 can be eliminated by reader R_3, duplicate reports of tags T_1 and T_2 will be forwarded by both readers R_3 and R_5 and the duplicates eliminated only by R_6. This generates 4 transmissions, instead of only 2.

cellphones) and is illustrated in Fig. 2(a). Such readers may locally detect tags and then report them to the host system using their cellular data link. Readers are assumed to be unable to communicate with each other, except through the host system. The second model considers the deployment of WLAN, Bluetooth, or 802.15.4 readers, that are able to locally communicate.[3] In this model, depicted in Fig. 2(b), only a subset of the readers communicate directly to the host, forming a backbone of the reader network. The other readers establish ad-hoc paths to backbone readers and forward tags reported by other readers. In this model, redundant tag reports may be more damaging than in the first model, since redundant tags may be propagated several hops before being eliminated (see Fig. 2(b) for an example).

2.2. Tree walking: An algorithm for detecting tags in the presence of collisions

RFID readers have the ability to accurately read (and report) data stored at multiple tags placed within their interrogation zone. Since readers broadcast their queries, multiple tags may end up responding simultaneously, leading to interference and inaccurate decoding of responses. This is known as the *tag collision problem*. Note that tag collision and reader collision are different problems.

Several techniques have been proposed to solve the tag collision problem. A popular solution, known as the tree walking algorithm (TWA) [27] is based on recursive traversal of the binary name tree of tag identifiers. The reader initially sends a broadcast query containing the "0" string. All tags in its interrogation zone whose identifier starts with a "0" bit reply. If a reply is received, or a tag collision is detected, the reader recurses on both subtrees 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 tag identifiers.

While TWA was designed to accurately read tag identifiers, it can also be used to read arbitrary data stored on tags, as long as the data's bit length is the same across all tags.

2.3. Reader collision: Problem definition

The tag collision problem occurs when one reader simultaneously interrogates multiple tags. In large scale environments, such as retail spaces, the interrogation zone of a single reader may not cover all the tags. Furthermore, tags can be randomly placed or moving. In such cases, accurate detection of all tagged objects requires (possibly random) deployment of multiple readers. A good reader placement ensures that the entire tag deployment area is covered by the union of the interrogation zones of all readers. Complete coverage implies coverage redundancy, since regions may be covered by the interrogation zones of multiple readers.

Redundantly covered tags may be incorrectly detected or even escape detection if readers with overlapping interrogation zones simultaneously query tags (possibly part of the tree walking algorithm). This scenario, known as the reader collision problem [12], is due to reader signal interference occurring at tags. Avoiding reader collisions is an essential requirement for error-free operation of an RFID system. We now formally define this problem. In the following, we use $|s|$ to denote the bit length of binary string s and $|\mathcal{A}|$ to denote the cardinality of set $\mathcal{A}$.

Let $S = (\mathcal{R}, \mathcal{T}, C, \beta)$ denote an RFID system. We use $\mathcal{R}$ to represent the set of readers and $\mathcal{T}$ the set of tags. Let $\psi = |\mathcal{R}|$ be the number of readers and $\theta = |\mathcal{T}|$ be the number of tags in the system. $C : \mathcal{R} \to \mathcal{T}^*$ is the coverage function, such that for any $R \in \mathcal{R}$, $C(R) \subseteq \mathcal{T}$ denotes the set of tags that are within the interrogation zone of R. β is the bit length of tag identifiers. We denote the set of all binary strings of length at most β with B^β. We use the notation $s \trianglerighteq s'$ to denote that binary string s is a prefix of binary string s'. Note that we do not make any assumption on the interrogation zone radii of readers. We discretize time as the set $\mathcal{U}$ of successive time frames, where the length of a time frame is

[3] Many existing cellphones are equipped with Bluetooth and/or WLAN communication capabilities.

sufficient for a query to propagate from a reader to any tag within its interrogation zone.

Definition 1 (*Unique Prefix Set of a Tag*). Given an RFID system $S = (\mathcal{R}, \mathcal{T}, C, \beta)$, let s_T denote the binary string that uniquely identifies T. Then, for any $R \in \mathcal{R}$ such that $T \in C(R)$, the unique prefix set for T with respect to R is defined as $P(T, R) = \{s \in B^\beta : s \trianglerighteq s_T, s \not\trianglerighteq s_{T'} \forall T' \in C(R)\}$.

The unique prefix set of a tag contains the set of prefixes that resolve tag collisions for that tag. Indeed, TWA successfully recognizes a tag when the minimum length string in its unique prefix set is queried. We now define the problem of scheduling reader queries to avoid reader collisions in addition to tag collisions.

Definition 2 (*Reader Collision Avoidance Problem*). Given an RFID system $S = (\mathcal{R}, \mathcal{T}, C, \beta)$, determine a function $t : \mathcal{R} \times \mathcal{T} \times B^\beta \to \mathcal{U}$ with minimum range, such that for any reader $R \in \mathcal{R}$ and any tag $T \in C(R)$, there exists $s \in P(T, R)$ such that $t(R, T, s) \neq t(R', T, s')$ for all $s' \in B^\beta$ and $R' \in \mathcal{R}$ satisfying $R' \neq R$ and $T \in C(R')$.

Informally, the above definition states that the goal of reader collision avoidance is to determine a mapping from each reader, each tag within its interrogation zone, and at least one prefix that uniquely identifies the tag to a time frame when it is safe for the reader to query the bit string. A reader query is said to be safe in the absence of transmissions from any other readers covering the corresponding tag during the same time frame.

We formally define the optimal tag coverage problem, which is the focus of the second part of this paper. To do this, we first formalize the notion of a redundant reader:

Definition 3 (*Redundant Reader*). Given $R \in \mathcal{R}$, if for all $T \in C(R)$, there exists $R' \in \mathcal{R}$ such that $T \in C(R')$, then R is said to be *redundant*.

The optimal tag coverage problem corresponds to the problem of finding the smallest set of readers that cover all the tags in the system. Clearly, in such a set of readers, no reader is redundant.

Definition 4 (*Optimal Reader Coverage Problem*). Given an RFID system $S = (\mathcal{R}, \mathcal{T}, \text{Cov}, \beta)$, determine the minimal size set $M \subseteq \mathcal{R}$ satisfying the following properties.

(i) For any tag $T \in \mathcal{T}$ there exists a reader $R \in M$ such that $T \in C(R)$.

(ii) There exists no redundant reader $R \in M$, that is, the set $M - \{R\}$ no longer satisfies property (1).

According to Definitions 3 and 4, all readers except R_1 in Fig. 1 are redundant, however, the set $M = \{R_1, R_2\}$ is the reader set providing optimal coverage.

A more general problem consists of minimizing the number of redundant tags that are reported by an RFID system. It would be desirable to have each tag reported only once, effectively decreasing the number and duration of reader wireless transmissions. We formally define this problem.

Definition 5 (*Optimal Tag Reporting Problem*). Given an RFID system $S = (\mathcal{R}, \mathcal{T}, \text{Cov}, \beta)$, determine a function $r : \mathcal{T} \to \mathcal{R}$ such that $T \in C(r(T))$ for all $T \in \mathcal{T}$.

Informally, a solution to the optimal tag reporting problem maps each covered tag to exactly one reader among those that cover it. Clearly, such a mapping is facilitated by global knowledge on the topology of tag distribution and reader deployment. However, in the absence of inter-reader communication, a solution to this problem is not straightforward. Observe that the optimal tag coverage problem may be considered a variation of this problem, in which the range of function r is minimized.

2.4. Network model

The algorithms presented in this paper are designed under the following conservative assumptions. Any relaxation of these conditions will only improve the performance of our algorithms.

- We make no assumptions on the number of readers and tags, or on the the underlying reader deployment or tag distribution topology. We do not assume the presence of a centralized entity capable of collecting information about the topology of the reader network or controlling the behavior of individual readers. Thus, our algorithms do not rely on inter-reader communication capabilities.
- We assume the presence of passive tags only, as opposed to active tags (the latter are more powerful but also more expensive). Passive tags use the energy of signals received from readers in order to respond to their queries.
- Tags have limited memory. Part of a tag's memory is read-only, used to store a unique identifier, and part of it is writable. In Section 2.1, we list a few types of tags with a writable memory of between 112 and 1000 bytes. Moreover, tags are capable of performing comparison and prefix matching. Prefix matching is a requirement of the tree walking algorithm (see Section 2.2) and the circuits performing comparison are similar in functionality and complexity.

Furthermore, we assume that any reader is able to detect tag collisions, occurring when multiple tags respond to one of its queries. In fact, this is a basic assumption necessary for any algorithm that resolves tag collisions and is a requirement for the tree walking algorithm described in Section 2.2.

3. Related work

We briefly describe related results on various aspects of the problems investigated here.

3.1. Collisions in RFID systems

The reader-collision problem in RFID systems was first documented in [12]. The solution proposed, of allocating different frequencies to interfering 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 readers. Furthermore, our solution guarantees with high probability (w.h.p.) that each reader is able to correctly read all the tags placed in its interrogation zone.

Another decentralized solution for the reader collision problem was proposed by the same group (Waldrop et al. [31]) shortly after. They proposed two decentralized MAC protocols for reader networks whose purpose is to allocate disjoint time slots (colors) for reader transmissions. The first protocol is the Distributed Color Selection (DCS) algorithm and the second one is Colorwave, or the Variable-Maximum Distributed Color Selection (VDCS) algorithm. The DCS/VDCS protocols are based on the existence of an interference graph whose links denote interference between the end-points corresponding to readers. That is, each reader is assumed to have knowledge of the other readers with whom its transmissions interfere. This knowledge does not restrict itself to identification data, but also requires interfering readers to be able to communicate with each other. In DCS, a reader that needs to transmit does so only during its color timeslot. If a collision occurs, the transmission is aborted and the reader changes color (timeslot). Furthermore, it reserves the newly chosen color by contacting all its neighbors (from the interference set) and

asking each to (randomly) choose a new color if there is a color conflict. DCS assumes a maximum value for the number of colors, which however proves detrimental if the probability of readers transmitting changes in time. The VDCS algorithm addresses this problem by dynamically allowing nodes to change the maximum number of colors.

The problem of the DCS/VDCS protocols is that they rely on the interference graph. Building this graph is a difficult problem in itself. One way of building it would be to perform manual RF site surveys, however, this solution requires significant human intervention. Another solution would to schedule reader transmissions such that at any time only two readers transmit and the rest listen. At the end of the process, each reader communicates interference data to the pairs of readers whose transmissions interfered. This solution is time consuming and transmission intensive ($\binom{n}{2}$ pairs of readers). It also requires a centralized deployment of the schedule. Finally and most importantly, the interference data would be incomplete, since the interference would only be measured at reader locations, instead of tag locations.

Ho et al. [14] have similarly studied the problem of minimizing reader collisions. The solution proposed, HiQ, learns the collision patterns of readers and assigns frequencies over time such that neighboring readers do not experience collisions. They made the observation that Colorwave, proposed by a subset of the authors of HiQ, while being able to dynamically adjust to changes in the environment, was unable to take advantage of the coordination and optimization that would be possible using a global view of the network. However, since a centralized solution offers a single point of communication and failure, HiQ is hierarchical, allowing for greater scalability and fault tolerance. Specifically, each reader communicates with a single "R-server". R-servers are responsible for preventing collisions between the readers that report to them. Each reader is assumed to be able to detect collisions with neighboring readers, then communicate the number and type of collisions detected (either frequency or tag collisions) and its successful reads to its R-reader. An R-reader feeds the collision patterns learned from readers into a dynamic on-line algorithm that then allocates resources (time slots and frequencies) back to the readers. R-servers are allocated a limited number of resources by a "Q-server", which is he highest element of the HiQ hierarchy. Each R-server communicates with a single Q-server, requesting additional resources or reporting collisions. Q-servers may e hierarchically organized, with a single root Q-server. Each Q-server allocates resources to the Q-servers and R-servers that are below it in the hierarchy, using a dynamic Q-learning algorithm. The algorithm infers collision constraints between readers based on reported collision information and from the requests for additional resources. While HiQ is decentralized, it relies on additional hardware, namely the R-servers and Q-servers. Our algorithm, RCA, makes no such demands and also requires much less communication with the backend system.

The Slotted Scheduled Tag Access protocol (SSTA) of Zhou et al. [36] is a hybrid between spatial, time and frequency division MAC protocols for RFID systems. It assumes a planned deployment of the reader network, where it is possible to perform RF site surveys and measure reader location and interference patterns. Moreover, once this information is collected, it is processed at a central location and each reader is assigned its resulting schedule. The problem is first studied under the assumption that no tag spatial distribution is known. The solution proposed greedily activates a set of non-interfering readers in each time slot, such that a maximum possible amount of new area is covered in each slot. When the spatial distribution of tags is known, the solution is made more effective by the fact that the time to read all tags in a given area can be easily approximated. Note that the authors propose solutions to the problem both in the single channel and in the multi-channel scenarios. The problems of the SSTA protocols stem from their limited applicability to models where the location and interference patterns of readers are known to a central computing facility. They cannot be applied to scenarios where this information is difficult or too expensive to collect.

Note that both Colorwave and HiQ are dynamic protocols, in the sense that they take as input read requests from readers and then try to allocate resources (slots or frequencies) on the fly to the requesting readers and those interfering with them. The SSTA algorithm is static, since it uses information collected from the deployed network in order to once and for all determine reader resource assignments. Our protocol, RCA, is a randomized protocol, where each reader attempts to perform a read operation multiple times at randomly chosen time slots. Thus, RCA is a dynamic algorithm, which however requires no assistance from other servers or readers.

Deolalikar et al. [11] take an ingenious, graph-theoretic approach at the problem of assigning resources (time slots or frequencies) to readers whose transmissions interfere. The input to their solution is also an interference graph of the reader network. The authors make the observation that a bipartite interference graph is the easiest to schedule, by firing for a read operation first the readers in one partition and then the readers in the second partition. When more resources are available (e.g., time slots or frequency channels) k-partite graphs are a better alternative since they are easier to derive from a reader network. The problem then is to efficiently transform an interference graph into a k-partite (or bi-partite) graph. The authors make the observation that this transformation can be performed in a succession of steps, where each step is either (i) an elimination of a collision constraint by moving the endpoint readers apart or (ii) an elimination of a reader. They then propose heuristics for choosing the best edge or vertex for removal from the interference graph, based on a measure called "correlation" of readers. The correlation of two readers consists of the number of tags covered by both readers. Namely, for edge removals, first the edges between readers with the highest correlation are removed. For vertex removals, first the readers with highest correlations with their neighbors are removed. Such readers are essentially what we call in this paper redundant readers. Thus, besides the aforementioned problem of requiring the existence of an interference graph, this solution attempts to solve the redundant reader problem (or at least count the tags covered by each reader and build a correlation table for each neighbor) before solving the reader collision problem, that is, before being able to read the tags. The authors do not provide a solution for building neither the interference graph nor the reader correlation values.

Birari and Iyer [6] proposed Pulse, a protocol that addresses reader collisions through the use of a "control" channel. Since the interference range of readers exceeds their transmission range, Pulse requires readers to send control data at a higher power, in order to allow for inter-reader communication. With this capability, the Pulse protocol allows readers to communicate with other readers with whom they are interfering. Then, readers can reserve the channel for a tag reading session, by periodically sending a beacon. If a reader does not receive beacons for a certain interval (constant time + random interval) it can signal its intent to read tags by sending beacons. Using a higher power for transmissions over the control channel also generates a much larger interference range, which can disturb the tag reading processes of readers that are even farther away. Moreover, this process also consumes more power and power consumption increases superlinearly with the transmission range.

3.2. Coverage problems in sensor networks

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 [30] 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 [34] 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. [33] 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 [29] 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. Carbunar et al. [7] proposed the use of Voronoi diagrams to determine in a distributed and localized manner the redundancy of sensors. Sensors only need the knowledge of their Voronoi cells, including the identities of the sensors that are their Voronoi neighbors. All of the above results use 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 reader network. Furthermore, we define coverage only in terms of the set of discrete points, tags. While this approach has the potential to discover more redundant readers, the problem is complicated by the scarce resources of tags.

3.3. Medium access control issues

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 [1]. 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 a reader is to detect *all* the tags in its interrogation zone. Such a reader does not know the identities of the tags and thus cannot send individual RTS messages. Moreover, the simultaneous reception of CTS messages initiated by tags leads to tag collision problems. Carrier sensing multiple access with collision detection (CSMA/CD) [8], 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 to detect remote collisions.

4. Reader collision avoidance

In this section 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 [27] (see Section 2.2). Our solution can be extended to work with other tree-based protocols such as the binary tree protocol [15,17,27] or the query tree protocol [21,35], which continuously split a set of tags into two subsets until each set has only one tag. We prefer TWA for its simplicity of exposition. Similar to TWA, a reader running RCA sends a broadcast query containing a certain prefix expected to match the identifiers of tags in its interrogation zone. However, unlike TWA where the lack of an answer is considered to denote absence of matching 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 with high probability (w.h.p.) that the choice of a time frame is not picked by another reader, thus alleviating the impact of reader collisions.

A reader divides time into disjoint epochs and each epoch is further divided into multiple disjoint time frames. The duration of a time frame is set to exceed the time a reader query needs to reach a reader's entire interrogation zone, plus the time a tag needs to process a query (one string comparison) and the time a tag answer needs to reach back the reader. During each epoch, a reader picks a frame uniformly at random and sends its query in that frame. This ensures that the queries of any two, potentially interfering readers will collide only if both readers choose the same time frame during the current epoch. If no tag answer is received, the reader repeats the query in a randomly chosen time frame of the next epoch. If a reader collision at matching tags has occurred during the query, the query duplication correlated with the random backoff decreases the chances of repeated reader collisions.

This is illustrated in Fig. 3. In Section 4.1, we prove that if the number of time frames per epoch is equal to the number of readers, ψ, and a query is not answered $O(\log \psi)$ times, then w.h.p., there are no tags matching the query in the interrogation zone of the reader. If, however, an answer is received, either as a clear tag response or by detecting a tag collision, in the next epoch the reader recursively moves to the next query, as in the TWA algorithm.

Note that readers cannot and do not need to detect tag collisions. If after a query a reader receives no reply, it means that either (i) there is no tag matching the query's prefix string or that (ii) a collision has occurred. After a single query without a reply, the reader cannot differentiate between these two cases. After $\log \psi$ queries, if still no answer has been received, the reader can conclude with high probability the fact that no tag matching the query lies in its interrogation zone. This result shows that having tag collisions for all of these queries is highly improbable, indicating that the tag collision problem will be avoided without having to detect any collision.

The readers need then to know the value of ψ in order to set the number of frames per epoch to the appropriate value. Since the readers are deployed by a single entity their number may be provided to each reader before deployment. If new readers are later deployed or existing readers have been detected as inactive (due to exhausted battery or physical faults) readers may be provided by the central host with new estimates of the number of readers. Depending on the network model (see Section 2.1), this operation can be performed either over the cellular link between the central host and individual readers or over the cellular link for the backbone readers who then propagate this information to the other readers using inter-reader communication capabilities.

The choice of using ψ frames per epoch and of repeating a query $O(\log \psi)$ times is made under the conservative assumption that all readers interfere with each other at all tags. Hence, the bound that we provide is the worst case bound with respect to the distribution of tags and deployment of readers. However, this is not always the case. In our experiments (see Section 7), we show that in realistic scenarios of random deployment of readers and tags, much fewer frames per epoch and repetitions per query are needed in order to

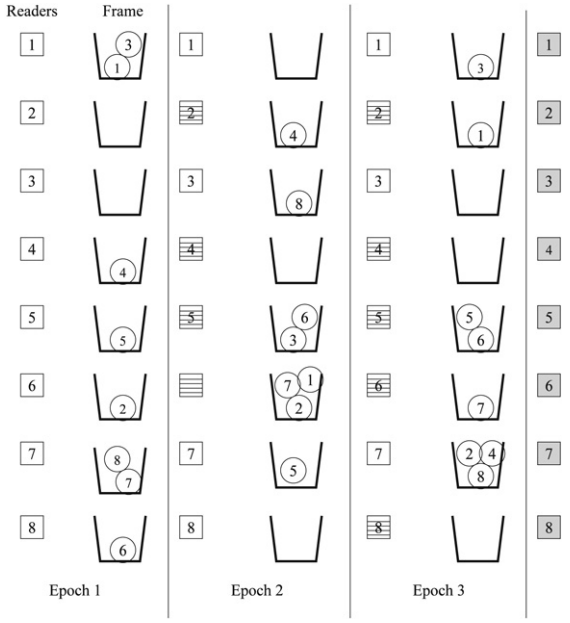

Fig. 3. Illustration of the RCA algorithm —squares denote readers, buckets denote time frames and labeled balls (the label indicating the source of the ball) denote queries sent by readers during different time frames. In each epoch, readers that have successfully transmitted information to a tag are 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 4.1 for a bound on the number of epochs) all readers are successful.

allow readers to accurately detect tags placed in their interrogation zones. Specifically, in one experiment we deployed 500 readers and 4000 tags and then increased the number of time frames per epoch from 1 to 38 ($\approx 4 \log \psi$). We have observed that when the number of frames per epoch was 9 ($\log \psi$) an average 99.3% of tags were discovered by readers. However, when the number of frames per epoch was 18 ($2 \log \psi$) each reader correctly read *all* the tags in its interrogation zone. We note that the number of repetitions per query was only 9 ($\log \psi$). We have subsequently used these values ($2 \log \psi$ frames per epoch and $\log \psi$ repetitions per query) in all our experiments (up to 1000 readers and 8000 tags) and our result held (all readers correctly discovered all their tags).

Implementation: Algorithm 1 presents the pseudocode for RCA using an Orca [3] 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).

Algorithm 1 The generic reader and tag behavior. `getRandom(v1, v2)` returns a random integer value between `v1` and `v2` 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.          exit;
27.       od
28.       guard getTime() - T ≥ T_out do
29.          sleep(n - frame - 1);
30.       od
31.    od
32. end
```

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. The reader can then safely recurse on the two children of the employed prefix (lines 23–26).

4.1. Analysis

We present an analysis of RCA based on two fundamental abstractions in randomized algorithms, *viz.* a balls and bins abstraction and the coupon collector paradigm. The balls and bins abstraction for RCA is used to analyze the distribution of readers that successfully communicate with a tag in a single epoch and is illustrated in Fig. 3. The coupon collector paradigm establishes the number of epochs required for all readers to successfully communicate with all tags.

As defined in Section 2.3, let ψ be the total number of readers, γ be the total number of tags in the system, and β be the bit size of tag identifiers. Let τ denote the number of time frames in each epoch. Our goal is to evaluate the number of epochs per query required to guarantee w.h.p. the success of the query. To establish an upper bound, we assume that interrogation zones of all ψ readers share

all γ tags. Note that this is a worst case assumption. 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 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 [23]. 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 $\psi e^{-\frac{\psi}{\tau}}$. ■

The coupon collector problem is defined as follows: there are n unique types of coupons and one is interested in collecting all types. In one trial, a coupon is chosen from the set of coupons, where the probability of acquiring each type is $1/n$. Then this coupon is placed back into the set, and the process is repeated. In this model, each reader corresponds to a unique coupon type. A tag acquires a coupon if and only if the corresponding reader is the only reader to query this tag. Hence, a reader successfully communicates with a tag if the tag acquires the coupon corresponding to the reader in question in at least one of the epochs. We are interested in the number of epochs required to ensure that all readers successfully communicate with all targeted tags with high probability. Consequently, our analysis of the RCA algorithm is based on the following fundamental result on the number of coupons that need to be collected to ensure that all coupons are acquired [23].

Theorem 1. *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* $\mathtt{O}(\log \mathtt{n})$.

We now show that the coupon collector paradigm provides an upper bound on the number of epochs required for all readers to communicate with all tags.

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

Proof. Modeling readers with coupons and each epoch with one step of coupon collection, we observe that RCA is a modified version of the coupon collector problem. In the classical coupon collection process, coupons are chosen one by one with replacement. In RCA, on the other hand, multiple coupons may be chosen at once (*i.e.*, without replacement). Lemma 1 provides the average number of coupons acquired in one step, $\psi e^{-\frac{\psi}{\tau}}$, which is equal to the average number of readers that successfully communicate with their target tags in a single epoch. After each epoch, all coupons are replaced and the process is repeated, as in classical coupon collector. Observe that this modification only increases the rate at which the coupons are acquired. In other words, if all parameters are fixed, then the probability of having a reader successfully communicate with its target tag in RCA is at least the probability of acquiring the corresponding coupon in one step of coupon collector. Consequently, the number of epochs required for each reader to send its query during a time frame is at most the number of samplings required in the classical coupon collector process. ■

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

Theorem 2. *If the number of time frames per epoch is set to the number of readers,* ψ, *then RCA requires only* $\mathtt{O}(\log \psi)$ *query repetitions to ensure w.h.p. the receipt of a reader's query by the target tags in its interrogation zone.*

Proof. If x is the number of query repetitions, using Lemma 1, Theorem 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 determine the time complexity of RCA.

Lemma 3 (*Complexity of RCA*). *The total number of epochs in RCA,* $\mathtt{T}_{\mathtt{RCA}}$, *is* $O(\gamma \log \beta \log \psi)$.

Proof. Since each reader covers γ tags of bit size β, the number of queries is $O(\gamma \log \beta)$. As each query is repeated $O(\log \psi)$ times by Theorem 2, we have $\mathtt{T}_{\mathtt{RCA}} = O(\gamma \log \beta \log \psi)$. ■

4.2. Discussion

Synchronization: An important observation is that time synchronization is not required across 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, the bounds presented above hold for an asynchronous system.

Topology of readers: For ease of analysis, we assumed that the interrogation zones of all readers intersect with each other, i.e., each tag is the interrogation zone of all readers. 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 matching 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 of the content of the queries, by abstracting queries sent at certain time frames as balls being thrown into bins.

Exponential back-off: A plausible improvement of RCA may use 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 2, 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

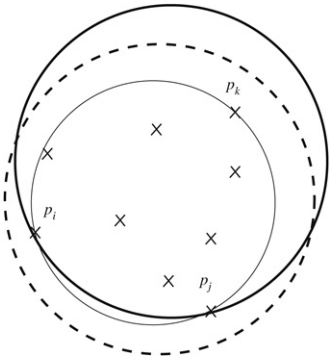

Fig. 4. Set of points covered by a circle of radius ρ, shown with an interrupted perimeter. There is a circle of radius ρ going through points p_i and p_j and covering all the other points. Shrink this circle until it first touches one more point, p_k. The resulting circle, has radius less than or equal to ρ.

constant epoch sizes, instead of exponential epoch sizes, generally used in Ethernet and other exponential back-off approaches.[4]

5. The optimal tag coverage problem

We first show that the optimal tag coverage problem is NP-hard even in a setting where global information about the distribution of tags and deployment of readers is available. Although this problem is closely related to the minimum set cover problem, a reduction from minimum set cover is not straightforward since the coverage sets of readers are constrained by the geometry of the two-dimensional Euclidean space. In the following, we show that in spite of this constraint, the problem is NP-hard, by reduction from the geometric disc cover problem. We first provide the following lemma, illustrated in Fig. 4.

Lemma 4. *Given a set of n points, $p_1, p_2, \ldots, p_n$, placed inside a circle of radius ρ, 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 $c(o_{ijk}, \rho)$. o_{ijk} is the mass center of p_i, p_j, p_k and $c(x, \rho)$ denotes the circle centered at x with radius ρ.*

Proof. We provide a constructive proof. If all the points are covered by a circle of radius ρ, then a circle of radius ρ going through 2 of the points and covering all the other points exists (see Fig. 4). 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 ρ, is the circumcircle of three of the n points, and covers all the other points. ■

We can now prove the following important result.

Theorem 3. *The optimal tag coverage problem is NP-hard.*

Proof. By reduction from the geometric disk cover (DC) problem, known to be NP-hard [13]. The input to the DC problem consists of a set of m points and a value ρ. The output is the minimum number of disks of radius ρ that cover all the points. We use the following polynomial time reduction from DC to the optimal tag coverage problem. Add a disk of radius ρ centered at each point in the input set of DC. Then, for all combinations of 3 points in the input set, add a disk of radius ρ, centered at the mass center of the 3 points. Let Σ denote the set of all disks created. The points input to DC form the input tags, and the disks in Σ form the input reader interrogation zones for the tag coverage problem. The reduction has $O(m^3)$ complexity. It is clear that the disks in Σ cover all the input points. Moreover, as a direct consequence of Lemma 4, the disks that form the solution to the DC problem are contained in Σ. Hence, the optimal solution to DC is also contained in Σ. Now consider the optimal solution for the tag coverage problem. If it does not correspond to an optimal solution for DC, then it should be possible to replace more than one of the disks in the solution to the tag coverage problem with a single disk of radius less than or equal to ρ. If so, the solution to tag coverage would not be optimal, since the 3-point disk derived from such a disk according to the construction of Lemma 4 is also in the input for the tag coverage problem. ■

6. Redundant reader elimination algorithm

We propose a distributed and localized algorithm for the redundant-reader problem. The algorithm is deterministic in nature, but it relies on the randomized RCA algorithm to avoid reader collisions. As specified in Section 2.4, we make no assumption on the topology of the reader network, effectively claiming no direct communication between readers. We assume, however, the existence of writable tags that are able to store information upon requests from in-range readers. We assume initially that RCA (see Section 4) has been previously executed by all readers to identify the tags in their vicinity. Later in this section, we discuss a simple modification to our algorithm to relax the synchronization assumption.

Centralized heuristics for RRE: We first motivate our RRE algorithm by considering a heuristic approach that assumes centralized knowledge of the network. Observe that the larger the number of tags a reader covers, the more likely it is to eliminate other readers. Therefore, similar to the greedy approximation algorithm for minimum set cover [9], a simple greedy heuristic for the tag coverage problem chooses the reader that covers the maximum number of tags to be turned on, step by step, until all tags are covered. Note that such an algorithm, GREEDY, assumes global knowledge on the number of tags covered by each reader. Since this information is not available for a realistic system, we approximate this greedy algorithm with a distributed and localized heuristic that assumes no centralized server or direct communication between readers.

Outline of RRE: The distributed RRE algorithm is based on readers marking each tag with a value, representing the highest number of tags covered by any reader in the tag's vicinity. The reader that issues the highest count for a tag, *holds* the tag. Therefore, rather than a centralized mechanism that chooses the reader with maximum tag coverage, each tag chooses the locally optimal reader in its vicinity. A reader holding none of its covered tags is declared redundant. Furthermore, a reader reports only the tags that it holds, providing a solution to the optimal tag reporting problem. The algorithm consists of two steps. In the first step, each reader attempts to write its tag count (number of covered tags) to all its covered tags. A 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 4.1, this process ensures w.h.p. that at least one write command issued by each reader is correctly received by all its covered tags. Thus, after $O(\log\ \psi)$ epochs, each tag stores the largest number of tags covered by a reader situated in its vicinity, along with the identity of that reader, called *holder* of the tag. Note that the RRE algorithm outlined here is deterministic, but it relies on the randomized RCA algorithm.

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

Algorithm 2 The generic 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; #is reader redundant?
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
```

In the second step, a reader queries each of the tags in its interrogation zone and reads the identity of the tag's holder. A reader that holds at least one tag is responsible for monitoring the tag and will have to remain active. However, a reader holding 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. Each read query issued by a reader for each of its tags is similarly repeated during random time frames for $O(\log\ \psi)$ consecutive time epochs to avoid reader collisions occurring at queried tags.

Implementation: Algorithm 2 formalizes our solution, which assumes writable tags. The functionality of a writable tag is shown in operation `run` of `WritableRFIDTag` (lines 4–13). The 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 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).

6.1. Analysis

In this section, we provide a proof of the accuracy of RRE and analyze its performance. The following theorem establishes the accuracy of RRE at the absence of reader failures or arrival of new tags, assuming that readers are synchronized. We elaborate on handling system update and synchronization issues in the next section.

Theorem 4. *RRE provides a feasible, but not necessarily optimal, solution to the optimal tag reporting problem, which is also a feasible, but not necessarily optimal, solution to the optimal tag coverage problem.*

Proof. According to Definition 5, a solution to the optimal tag coverage problem maps each covered tag to exactly one reader. In RRE a covered tag T is locked by only one reader R, of those that cover it. The reader R has the property that among all the readers covering T, R covers most tags. In case of a tie, the reader with the highest identifier value is chosen. ■

Lemma 5 (*Complexity of RRE*). *The total number of epochs in RRE,* T_{RRE} *is* $O(\gamma \log \beta \log \psi)$.

Proof. The complexity of RCA, is $O(\gamma \log \beta \log \psi)$ (see Section 4.1). The first step of RRE, where each reader sends a write command to all its tags, takes $e \log \psi$ epochs. The second step, where readers send queries to each of their tags, takes $\gamma e \log \psi$ epochs. Thus, $T_{RRE} = O(\gamma \log \beta \log \psi)$. ■

6.2. Discussion

Synchronization: We have assumed until now that all readers have already executed RCA, detecting all the 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 Fig. 1, the count of tag T_4 is 4, from reader R_2. However, if we assume that initially, 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 Fig. 1 for illustration): since R_4 only covers two 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 identifying itself to be the holder of T_4 and T_5, will also decide to stay active, in spite of its redundancy.

In order to solve this problem, we require readers to maintain a list of tags held, and to passively listen for tag responses to queries initiated by other readers. The duration of this phase is upper bounded by the time taken by a reader to get an estimate of the maximum number of tags a reader can cover. The listening phase proceeds as follows: assume that a reader R overhears a tag response to a query initiated by another reader and the query content is R_x, T_y, c (see Algorithm 2 line 11). This query indicates that the holder of tag T_y is R_x with a tag count c. Then, if c is larger than its own tag count and $R_x \neq R$, reader R removes tag T_y from its list of held tags. When the list is empty, the reader becomes redundant and can be safely turned off. Theorem 2 (see Section 4.1) 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

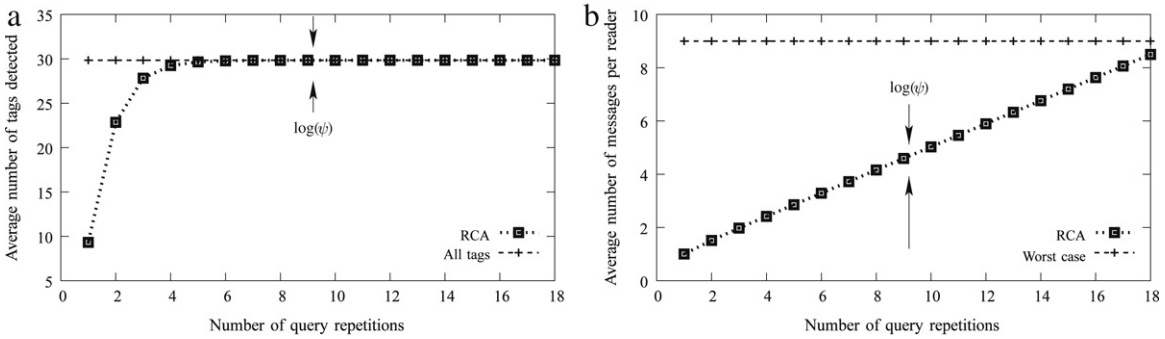

Fig. 5. Performance of RCA when the number of epochs per query grows from 1 to 2 log ψ, for a total of 500 readers and 4000 tags. The number of time frames per epoch is 2 log ψ, where ψ is the number of readers.

tag count. Using the example in Fig. 1, assume R_4 has T_4 and T_5 in its list of held tags on completion of its first step of RRE. During R_2's execution of the first step of RRE, R_2 chooses at least two time frames during e epochs, one for T_4 and T_5, when no other reader is transmitting. In these time frames, R_4 overhears the replies of T_4 and T_5. 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, tags and readers may fail and new components may be randomly deployed. Scenarios where new tags are deployed or existing tags move are particularly important. A simple extension of RRE that maintains the invariant of having each covered tag reported once, consists of periodically executing RRE. Let us denote the period using T_q. This simple solution has the following problem illustrated in Fig. 1. Assume that after executing RRE the first time, one of the active readers, R_2, fails. Then, when the rest of the readers execute RRE, tags T_2, T_4, and T_5 have the associated count value set to 4. Then, readers R_3 and R_4 discover inaccurately, their redundancy and do not report tags T_2, T_4 and T_5.

One solution to this problem requires each reader to execute RCA periodically, every T_q time units, to identify all its covered tags, including newly deployed ones. Subsequently, each reader resets the count value of each of its covered tags and re-executes RRE. A tag agrees to set its counter to a smaller value, 0, since 0 is a control value (a reader covering no tags will not issue a `write` command containing a 0 tag count field).

Another problem with this solution is illustrated through the following example: assume that when the second time R_2 executes RRE, it sets the counter of its tags first to 0 and then to 4. Assume then that R_3 and R_4 execute RRE after R_2. Then, R_3 sets the counter of T_2 to 0 and then to 1 and R_4 sets the counter of its tags first to 0 and then to 2. This will force both R_3 and R_4 to report their tags, leading to duplicate tag reports. A solution for this problem is to set the period T_q of each reader to be inversely proportional to the tag count of the reader. Then, R_2 executes this procedure more often than R_3 and R_4, eventually causing them to discover their redundancy. Note that if the tag count of R_2 changes, becoming smaller than that of readers R_1, R_3, and R_4, reader R_2 will become redundant.

Another solution relies on more complex tags equipped with timing circuits. 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.

7. Experimental results

In this section we experimentally investigate the performance of our algorithms. 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. All of our simulations are performed by deploying tags and readers uniformly at random in a 1000 units $\times$1000 units domain. We assume HF (13.56 MHz) tags with 64-bit identifiers, such as Tag-It HF-1 from TI, I-Code SLI from Philips, my-d SRF55VxxP from Infineon or LRI512 from ST Microelectronics. Instead of restricting our attention to a particular type of RFID readers, in the following we evaluate the performance of our algorithms in terms of various metrics of interest, in several deployment scenarios. Our goal is also to stress test our algorithms in settings consisting of large numbers of tags, readers or high tags per reader densities.

7.1. Performance of RCA

We experimentally analyze the accuracy and message overhead introduced by RCA, by comparing its performance with the simple tree walk algorithm [27](TWA) and with a simpler version of RCA, RCAv.1. In RCAv.1, each query is sent the maximum number of times, irrespective of the result of the query. Note that in RCA, a reader does not repeat a query if the result is a success, that is, it receives an answer from a tag or detects a tag collision.

In the first experiment we randomly place $\gamma = 4000$ tags and $\psi = 500$ readers, each with an interrogation radius of 50 units in the 1000 units $\times$1000 unit square area. We increase the number of repetitions (epochs) per query from 1 to 2 log ψ. Fig. 5(a) shows the average number of tags detected by a reader, compared with the average number of tags actually placed in the interrogation zone of the reader. When RCA is used, the number of tags discovered by a reader quickly converges to the number of tags placed in its interrogation zone. For 9 ($=\log\psi$) repetitions per query RCA allows any reader to discover *all* the tags placed in its interrogation zone. This shows that the result of Theorem 2 (see Section 4.1) is valid in a worst case scenario. Moreover, for a realistic distribution of readers and tags, the constant hidden in the big-oh notation is small, *i.e.*, is close to 1. Fig. 5(b) shows the corresponding number of messages per query generated by RCA in this scenario. As expected, the growth in the number of messages sent per query is linear in the total number of query repetitions. However, when each query is repeated up to 9 ($\log\psi$) times, RCA generates on average only half, 4.5 messages, of RCAv.1. Thus, in the following experiments we consistently repeat each query of RCA at most log ψ 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. 6(a) shows our observations. As expected, the detection accuracy of readers improves when longer epochs are employed, 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

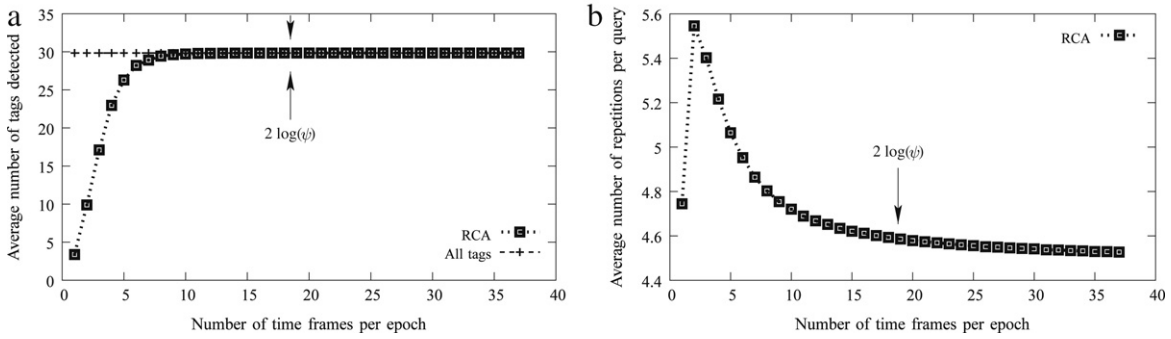

Fig. 6. Performance of RCA when the number of time frames per epoch increases from 1 to 38, for a total of 500 readers and 4000 tags. The number of time epochs per query is $\log \psi$, where ψ is the number of readers.

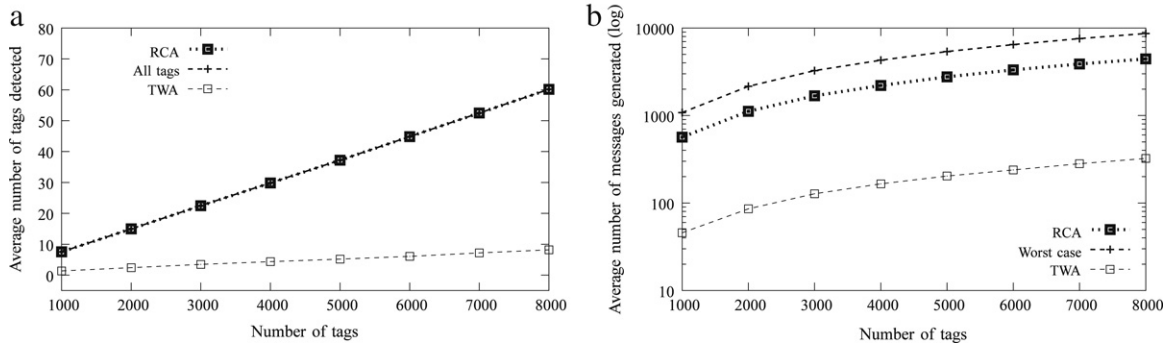

Fig. 7. Scalability of RCA and TWA when the number of tags increases from 1000 to 8000, while the number of readers is 1000.

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 $O(\log b)$ [23].

For the same scenario, Fig. 6(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, used in RCAv.1. Thus, by not repeating successful queries, our randomized algorithm saves, on average, more than 4 messages per query. This also shows that for realistic scenarios the necessary number of time frames per query can be significantly smaller than ψ, the value used in Theorem 2 (see Section 4.1). As a consequence, all the following simulations evaluate RCA using only $2 \log \psi$ time frames per epoch.

In the following experiment, we investigate the scalability of RCA in terms of the number of deployed tags, by randomly placing 1000 readers with an interrogation radius of 50 units and increasing 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. 7(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. 7(b) shows the corresponding number of messages per readers generated by the two algorithms, on a logarithmic scale, compared to the total number 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 for 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. 8(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. 8(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 to roughly 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. 9(a) shows the accuracy of RCA compared with TWA. RCA discovers *all* the tags until the interrogation radius reaches 85 units. 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 significantly 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. 9(b) shows the corresponding number of messages generated by the same algorithms. It confirms the results shown in Figs. 7(b) and 8(b). TWA generates only a fraction of the

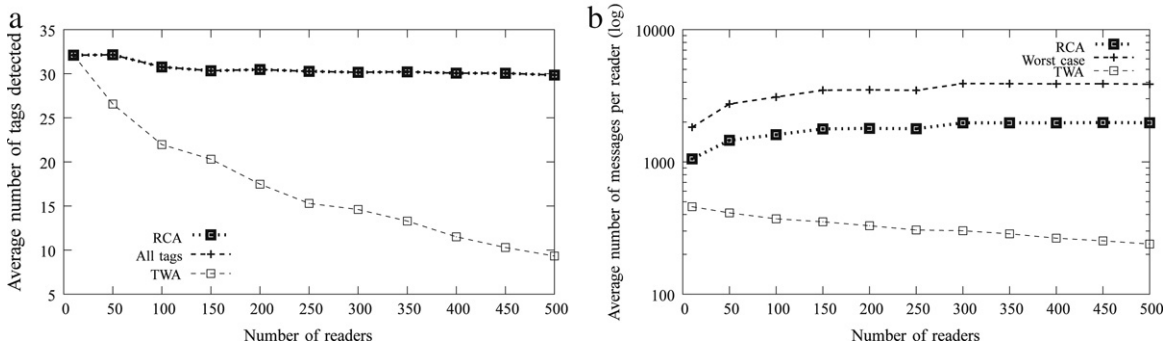

Fig. 8. Scalability of RCA and TWA when the number of readers increases from 10 to 500, for 4000 tags.

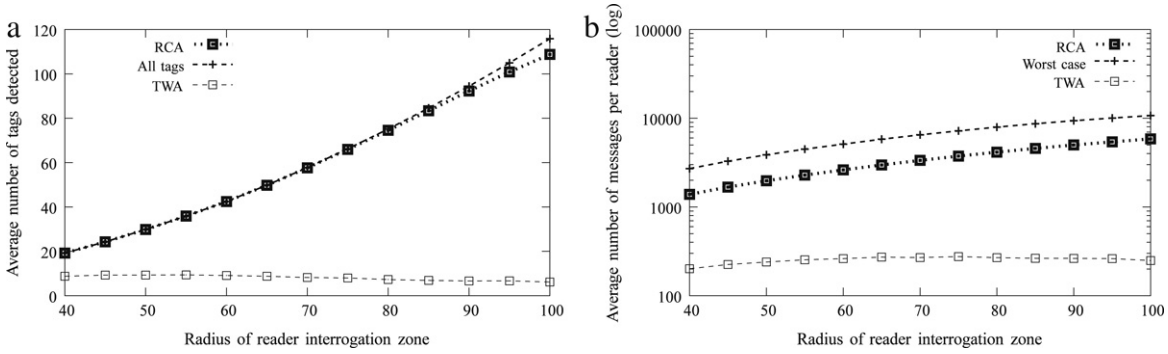

Fig. 9. Performance of RCA and TWA when the interrogation zone radius of readers increases from 40 units to 100 units, for 500 readers and 4000 tags.

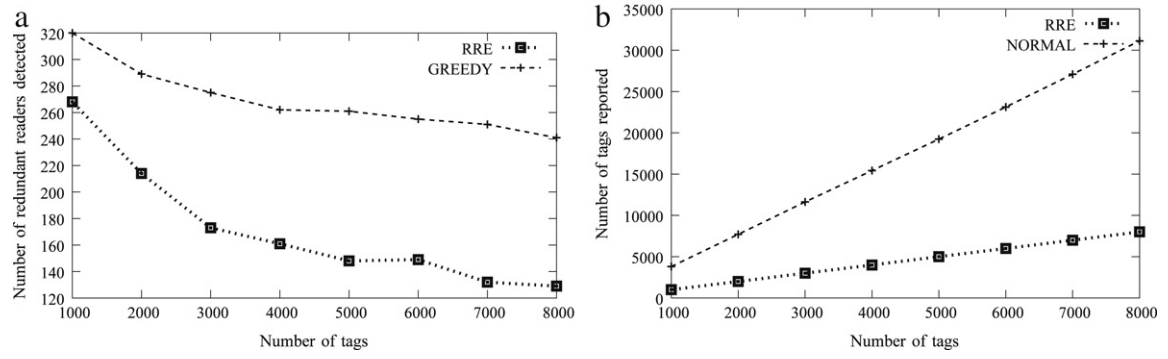

Fig. 10. Performance of RRE when the number of tags randomly deployed increases from 1000 to 8000. Throughout this experiment 500 readers are randomly deployed, each with an interrogation radius of 50 units. (a) Number of redundant readers discovered by RRE and GREEDY. (b) Number of tags reported by RRE and by NORMAL.

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.

7.2. Efficiency of RRE

We investigate the performance of our redundant-reader detection algorithm by comparing it with the centralized greedy approximation algorithm, GREEDY, described in Section 6. The performance of GREEDY is expected to be better than that of the proposed distributed and localized algorithm, RRE, since GREEDY makes all decisions based on global information. However, it should be noted that GREEDY is difficult and often impractical to implement, since it requires centralized knowledge of the reader topology.

We also compare the performance of RRE in terms of the total number of tags that are reported by the system, with a normal behavior of the reader network, denoted NORMAL, when no redundancy elimination algorithm is used. In the following experiments, readers and tags are randomly deployed, with the constraint that each tag is covered by at least one reader.

In the first experiment we randomly place 500 readers with a 50 unit interrogation zone radius and between 1000 and 8000 tags in the 1000 units $\times$ 1000 units domain. Fig. 10(a) shows the number of redundant readers discovered by RRE and GREEDY. For fewer tags deployed, RRE is reasonably close to GREEDY, by discovering 83% of the redundant readers discovered by GREEDY. As the number of tags increases, the performance of RRE relative to GREEDY degrades. However, RRE consistently discovers over 50% of the redundant readers discovered by GREEDY. Both GREEDY and RRE discover fewer redundant readers as the number of deployed tags increases. Both algorithms base their decision on the number of tags covered by readers. By increasing the tag density, the distribution of tags per reader becomes more uniform, making it more difficult to choose good, active 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 Fig. 11, where each of readers R_2, R_3 and R_4 covers four tags. While

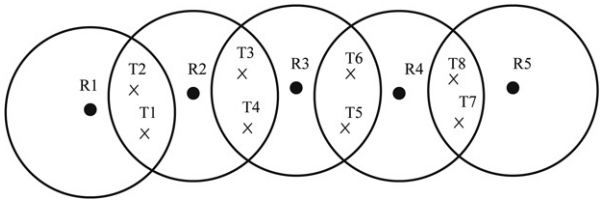

Fig. 11. Difficulty of consistently breaking ties. The optimal solution keeps only R_2 and R_4 active. However, in a scenario where R_2, R_3, and R_4, each covering 4 tags, hold a different set of tags, all of them will have to be active.

the optimal solution requires only R_2 and R_4 to be active, we can imagine a run of RRE where R_4 holds $T_5, \ldots, T_8$, R_3 holds T_3 and T_4 and R_2 holds 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 may 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 location.

For the same deployment scenario, Fig. 10(b) compares RRE with NORMAL. Our results show that RRE reports each tag exactly once. This is because each tag is associated with a single reader, the one that covers it and has the largest tag count. In this respect, RRE consistently performs almost 4 times better than NORMAL. This is expected to lead to a significant decrease in system-wide energy consumption.

The second experiment compares the performance of RRE and GREEDY when the number of randomly deployed readers increases from 50 to 1000, when the total number of tags is 4000. The interrogation zone radius of readers is set to 50 units. Fig. 12(a) shows the results of this experiment. For scarce deployment of readers, very few of the readers are redundant. As their density increases, however, so does the number of redundant readers. For example, for 1000 readers, GREEDY discovers almost 800 to be redundant. While initially RRE is very accurate, as the number of readers increases, RRE discovers fewer redundant readers. However, when between 500–1000 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 readers increases, the number of readers whose interrogation zones overlap, also increases, generating more contentions.

Fig. 12(b) compares RRE and NORMAL, for the same reader and tag deployment. When computing the average tag per reader for RRE, we do not consider the redundant readers, but only the ones that need to report. Our results show that for small reader networks, the average number of tags covered by a reader is high. This leads to a small difference in the number of tags reported by RRE and NORMAL. For instance, for a network of 50 readers, a reader running RRE reports roughly 80% of the number of tags it covers. However, for larger networks, this difference becomes more significant. Even though many of the readers are redundant, the average number of tags reported by an active reader running RRE can be as much as 3 times smaller than the average number of tags reported by a reader running NORMAL. For example, for a network of 1000 readers, 580 readers are discovered to be redundant by RRE and do not have to report tags. On average, each of the 420 active readers needs to report less than 10 tags. In contrast, NORMAL not only requires each reader to report tags, but on average, a reader reports more than 29 tags.

The final experiment measures the relationship between the number of redundant readers discovered by RRE and GREEDY and the interrogation zones of readers. We randomly deploy 500 readers and 4000 tags, and increase the interrogation radius of readers from 40 to 100 units. Fig. 13(a) shows that, as expected, with the increase in the interrogation radius of 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.

While the number of redundant readers discovered by RRE is smaller than GREEDY, the average number of tags reported by an active reader running RRE is quite small. Fig. 13(b) compares RRE and NORMAL in terms of the average number of tags reported by a reader. For RRE, the increase in this number is sublinear with the increase in interrogation zone radius. In contrast, the average number of tags reported by a reader running NORMAL increases superlinearly. Specifically, for an interrogation zone radius of 40 units, on average, one of the 400 non-redundant readers running RRE reports 10 tags, whereas an average NORMAL reader reports 20 tags. For an interrogation radius of 100 units, one of the 200 non-redundant readers reports approximately 20 tags, almost 6 times less than an average NORMAL reader, reporting 115 tags.

It would be interesting to understand the effects RRE's reduction in the number of tags reported by an average reader to the traffic generated by the reader network. For this, we consider Mica mote compatible readers [28] and tags with 64 bit identifiers. The TinyOS packet size is 36, with a payload of 29 bytes. We consider a random deployment of 4000 tags and 500 readers, each reader with an interrogation zone radius of 50 units. The tags and readers are deployed in the above mentioned 1000×1000 units region.

In this scenario, RRE discovers 161 redundant readers. None of the redundant readers has to report any tags. On average, one of the remaining 339 readers reports 12 tags, that is, send 4 TinyOS packets. Without using RRE, *each* reader needs to report tags.

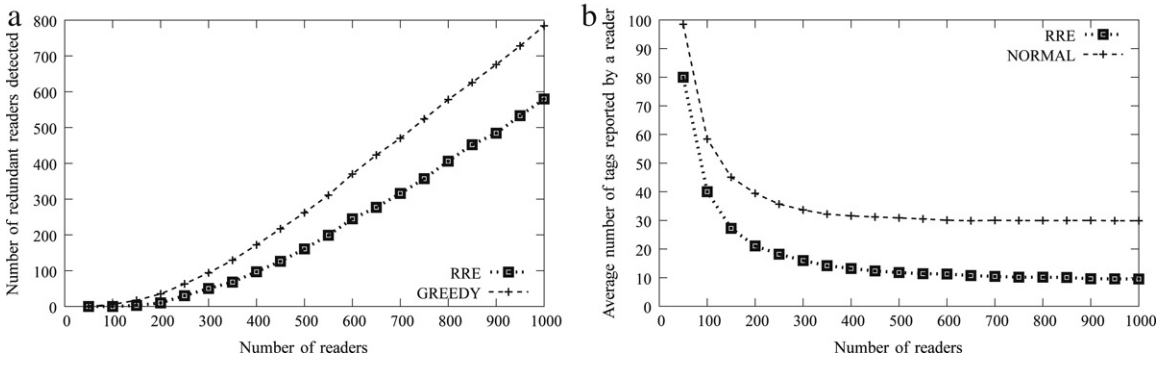

Fig. 12. Performance of RRE when the number of readers randomly deployed increases from 50 to 1000, for a total of 4000 tags. The interrogation radius of readers is of 50 units. (a) Number of redundant readers discovered by RRE and GREEDY. (b) Average number of tags reported by a reader running RRE versus NORMAL.

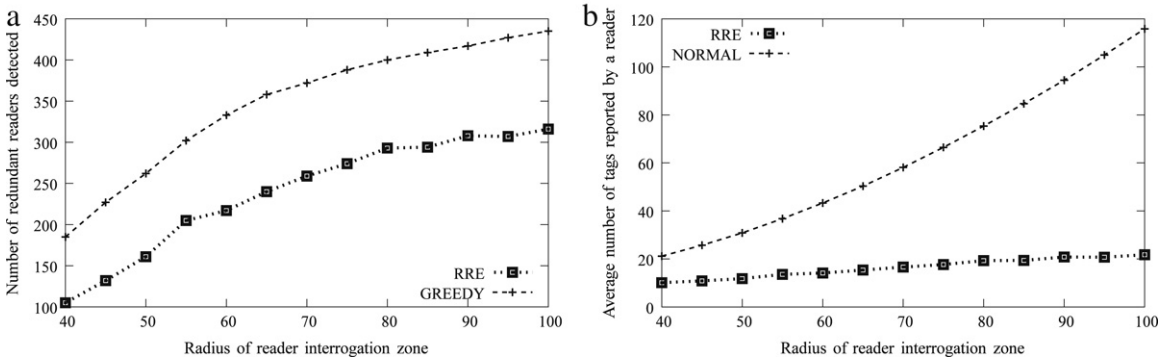

Fig. 13. Performance of RRE when the interrogation radius of readers increases from 40 to 100 units. Throughout the experiment, the number of readers is 500 and the number of tags is 4000. (a) Number of redundant readers discovered by RRE and GREEDY. (b) Average number of tags reported by a reader when running RRE versus NORMAL.

On average, one of the 500 active readers needs to report 31 tags, that is, send 9 packets. From a different perspective, when all readers execute RRE, a total of 1104 packets are sent. Without RRE, a total of 4257 packets are generated, almost 4 times as many!

8. Conclusions

In this paper we address two important problems in wireless RFID systems. The first problem, that of accurately detecting the tags covered by each reader, is made difficult by reader collisions occurring at remote tags. The second problem relates to extending the lifetime of the reader network by detecting and temporarily disabling the wireless interfaces of redundant readers. We define redundancy in terms of discrete sets of points, 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 tags. We provide a probabilistic analysis of the algorithms. Our extensive simulations show the impact of reader collisions on the accuracy of a tree walking algorithm [27] (TWA). Moreover, they show that our solution achieves high accuracy at the expense of slightly increased traffic overhead (on average 4 messages per query). We also show through simulation that our decentralized redundant-reader elimination heuristic is efficient when compared to a greedy approximation of the optimum solution, running with centralized knowledge of the reader and tag topology.

References

[1] N. Abramson, The ALOHA system: Another alternative for computer communications, in: AFIPS Conf. Proc., Fall Joint Computer Conf, 1970.
[2] Beth Bacheldor, Luggage Tag Has A Whole New Meaning. http://www.informationweek.com/story/showArticle.jhtml?articleID=16000658.
[3] Henri E. Bal, Raoul Bhoedjang, Rutger Hofman, Ceriel Jacobs, Koen Langendoen, Tim Ruhl, M. Frans Kaashoek, Performance evaluation of the Orca shared-object system, ACM Trans. Comput. Syst. 16 (1) (1998) 1–40.
[4] Baracoda. An efficient way to add RFID reader/encoder to Bluetooth PDA and mobile phones. http://www.baracoda.com/baracoda/products/p_21.html.
[5] Ann Bednarz, Wireless technology reshapes retailers, Network World, 12 August 2002.
[6] Shailesh M. Birari, Sridhar Iyer, Pulse: A mac protocol for RFID networks, in: EUC Workshops, 2005.
[7] Bogdan Cărbunar, Ananth Grama, Jan Vitek, Octavian Cărbunar, Redundancy and coverage detection in sensor networks, ACM Trans. Sen. Netw. 2 (1) (2006).
[8] Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications, Institute of Electrical and Electronics Engineers, 1996.
[9] V. Chvátal, A greedy-heuristic for the set cover problem, Math. Oper. Res. 4 (1979) 233–235.
[10] Crossbow Technology Inc. RFID and Asset Tracking. http://www.xbow.com/Industry_solutions/RFID.htm.
[11] Vinay Deolalikar, Malena Mesarina, John Recker, Salil Pradhan, Perturbative time and frequency allocations for RFID reader networks, in: Proceedings of Emerging Directions in Embedded and Ubiquitous Computing, 2006.
[12] D.W. Engels, S.E. Sarma, The reader collision problem, in: IEEE International Conference on Systems, Man and Cybernetics, 2002.
[13] R. Fowler, M. Paterson, S. Tanimoto, Optimal packing and covering in the plane are np complete, Inform. Process. Lett. 12 (3) (1981) 133–137.
[14] J. Ho, D.W. Engels, S.E. Sarma, Hiq: A hierarchical q-learning algorithm to solve the reader collision problem, in: Proceedings of the Applications and the Internet Workshops, 2006.
[15] D.R. Hush, C. Wood, Analysis of tree algorithms for RFID arbitration, in: Proceedings of ISIT, 1998.
[16] IATA. IATA Introduces RFID Standard for Baggage Tags. http://www.iata.org/pressroom/news_briefs/2005-11-18-01.htm.
[17] M. Jacomet, A. Ehrsam, U. Gehrig, Contactless identification device with anticollision algorithm, in: Proceedings of IEEE Conference on Circuits, System, Computers and Communications, 1999.
[18] Edwin Kalischnig, RFID: Making sense of sensor-based technology, Manufacturing and Logistics IT, July 2004.
[19] P. Karn, MACA—A new channel access method for packet radio, in: ARRL/CRRL Amatuer Radio 9th Computer Networking Conf., 1990.
[20] Olga Kharif, What's lurking in that RFID tag? http://www.businessweek.com/technology/content/mar2006/tc20060316_117677.htm?chan=technology_technology+index+page_computers, March 2006.
[21] Ching Law, Kayi Lee, Kai-Yeung Siu, Efficient memoryless protocol for tag identification, in: Proceedings of the International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, 2000.
[22] Mobile Magazine, Nokia 5140 RFID Reader. http://www.mobilemag.com/content/100/104/C2607.
[23] R. Motwani, P. Raghavan, Randomized Algorithms, Cambridge University Press, 1995.
[24] Mary Catherine O'Connor, RFID Journal, Reva Announces RFID Network Design. http://www.rfidjournal.com/article/articleview/1638/1/1/.
[25] PGS Electronics. Wireless RFID Systems. http://www.pgselectronics.com/PGSRFID.htm.
[26] RFID Gazette. HP Uses RFID Tags to Track Runners at Boston Marathon. http://www.rfidgazette.org/2004/05/hp_uses_rfid_ta.html.
[27] Sanjay E. Sarma, Stephen A. Weis, Daniel W. Engels, RFID systems and security and privacy implications, in: CHES'02, Springer-Verlag, 2003, pp. 454–469.
[28] SkyeTek, http://www.skyetek.com/readers_Mini.html, January 2004.
[29] Sasa Slijepcevic, Miodrag Potkonjak, Power efficient organization of wireless sensor networks, in: IEEE ICC, 2001.
[30] Di Tian, Nicolas D. Georganas, A coverage-preserving node scheduling scheme for large wireless sensor networks, in: Proceedings of the 1st ACM WSNA, ACM Press, 2002, pp. 32–41.
[31] J. Waldrop, D.W. Engels, S.E. Sarma, Colorwave: A MAC for RFID reader networks, in: Wireless Communications and Networking, WCNC, 2003.
[32] Wireless Dynamics, Wireless Dynamics Inc. announces the mini-SDiD. http://www.wdi.ca.
[33] Fan Ye, Gary Zhong, Songwu Lu, Lixia Zhang, Peas: A robust energy conserving protocol for long-lived sensor networks, in: 23rd IEEE ICDCS, 2003.
[34] Honghai Zhang, 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.
[35] Feng Zhou, Chunhong Chen, Dawei Jin, Chenling Huang, Hao Min, Evaluating and optimizing power consumption of anti-collision protocols for applications in RFID systems, in: ISLPED'04: Proceedings of the 2004 International Symposium on Low Power Electronics and Design, 2004.
[36] Zongheng Zhou, H. Gupta, S.R. Das, Xianjin Zhu, Slotted scheduled tag access in multi-reader RFID systems, in: Proceedings of ICNP, 2007.

Bogdan Carbunar is a senior staff researcher at Motorola Labs, where he is developing applications for mobile devices. He received a B.S. in Computer Science from Politehnica University of Bucharest, Romania in 1999, and a Ph.D. Degree in Computer Science from Purdue University in 2005. His interests include network security and applied cryptography, with applications in electronic payments and secure data outsourcing.

Murali Krishna Ramanathan received his Ph.D. in Computer Science from Purdue University in 2008. His research interests lie in Software Engineering, program analysis, distributed system and graph algorithms. He received the Maurice Halstead Award in the year 2006 for his research on applying sequence analysis to improve program correctness.

Mehmet Koyutürk is T. & D. Schroeder Assistant Professor of Computer Engineering & Networking, at the Electrical Engineering and Computer Science Department of Case Western Reserve University. He received his Ph.D. in Computer Science from Purdue University in 2006. His research interests include bioinformatics and computational biology, data mining and knowledge discovery, and algorithms for scientific computing and distributed systems.

Suresh Jagannathan is a Professor of Computer Science and University Faculty Scholar at Purdue University. Prior to joining Purdue, he was a Senior Director at Storage Networks, and a Senior Research Scientist at the NEC Research Institute. His interests are in programming languages and their implementation, distributed and concurrent systems, and software engineering. He received his M.S. and Ph.D. from MIT.

Ananth Grama is a Professor of Computer Science at Purdue University. Prior to joining Purdue as an Assistant Professor in 1996, Ananth received his Ph.D. at the University of Minnesota. His main areas of research include parallel and distributed computing, scientific computing, and large-scale data handling and analysis. He has (co)authored a number of papers and two textbooks on these topics. He is a recipient of the 1998 NSF CAREER award, was named 1999 Outstanding Assistant Professor of the School of Science, the 2002 School of Science Teacher of the Year at Purdue, and was designated a University Faculty Scholar in 2002.