Distributed Data Query in Intermittently Connected Passive RFID Networks

Zhipeng Yang, *Student Member*, *IEEE*, Ting Ning, *Student Member*, *IEEE*, and Hongyi Wu, *Member*, *IEEE*

Abstract—This paper focuses on distributed data query in intermittently connected passive RFID networks, which are characterized by extraordinarily limited communication capacity and asynchronous and opportunistic communication links. To address such unique challenges, we propose a distributed data query framework that clusters RFID readers and establishes a 0-1 Knapsack model based on dynamic packet appraisal to enable highly efficient data transmission. We implement a prototype by using Alien RFID gears and carry out experiments that involve 52 volunteers for 14 days to evaluate the proposed data query framework.

Index Terms—Distributed query, intermittent networks, passive RFID

1 Introduction

As one of the primary motivating applications of today's sensor networks, a diversity of wildlife studies have employed sensors for animal tracking and monitoring [1], [2], [3], [4]. However, a recent investigation reveals that the weight of the sensors must be under 5 percent of the weight of the animal; otherwise, the overweight additions often lead to high mortality rate of the animals being studied [5]. Consequently, small animals including 81 percent of bird species and 67 percent of mammal fauna cannot carry any active devices (such as GPS receivers, Crossbow sensors, and active RFID tags that are battery-powered). Despite continuous efforts to miniaturize sensors, the minimum weight of an active device is bounded inevitably by its power source and casing, where the former must be adequate to support desired communication range and lifetime, while the latter must be heavy duty for protecting power source and powered electronic circuits under harsh environment.

Aiming to address the strict weight constraints discussed above that limit the applicability of active sensors, the Featherlight Information Network with Delay-Endurable RFID Support (FINDERS) is proposed in [6], [7], by exploiting the ultralight, durable, flexible and battery-free passive RFID tags, for such long-lasting pervasive applications as tracking and monitoring of small wildlife. Furthermore, the rostering problem based on the FINDERS platform is discussed in [8], aiming to create a list of unique IDs that appear in the network.

1.1 An Overview of FINDERS

Fig. 1 illustrates an overview of the FINDERS system, consisting of fixed readers and mobile tags, each associated with a unique ID. The former are powerful devices, with large storage space, high computing power, and sufficient battery capacity. For instance, the Alien's ALR-9900 reader employed in our experiments possesses of 64-MB RAM and flash memory, plus an interface for extended computation and storage. The readers are often deployed according to specific applications. For example, they can be installed at "hot spots" or "choke points," where animals visit frequently or have to move through due to significant movement barriers otherwise. Experiments show that a car battery of $12\ \mathrm{V} \times 60\ \mathrm{Ah}$ can supply the reader for 35 hours under a scanning frequency of 1 Hz, sufficient to sustain continuous function of the reader in a wide range of applications when it is coupled with a suitable solar charger.

While readers are fixed, the thin and light tags are attached to moving targets (such as the wildlife being studied) and, thus, become mobile. Many off-the-shelf passive tags that meet the required weight constraint and support sufficient reading/writing range can be employed in FINDERS. For example, an Alien ALN-9540 tag adopted in our experiments measures $8.15 \times 94.8 \times 0.05\ \mathrm{mm}$, weights less than one gram, and achieves a communication distance of 20 ft. Since a tag has extremely limited storage space (e.g., the Alien tag can hold up to 20 bytes only), a block-based scheme is adopted in FINDERS to expand tag capacity by leveraging the aggregated space of multiple passive tags, achieving a total capacity of tens to hundreds of bytes. In this paper, we simply refer to a *tag* that can be a single tag or a block of tags.

Both reader and tag can be integrated with sensing elements for enhanced sensing capabilities [9].

1.1.1 Data Acquisition

FINDERS acquires a diversity of data depending on applications, including *meeting events*, *on-tag sensor data*, and *on-reader sensor data*. A reader periodically scans nearby tags. When a reader detects a tag (i.e., acquires a tag ID), it records a *meeting event*, including the IDs of the reader and the tag and their meeting time. During the meeting event, the tag is powered up by the reader. Its integrated sensing elements (e.g., enabled by the Wireless Identification and

• *The authors are with the Center for Advanced Computer Studies, University of Louisiana at Lafayette, Lafayette, LA 70504. E-mail: {zxy1767, wu}@cacs.louisiana.edu, txn6704@louisiana.edu.*

Manuscript received 20 Jan. 2012; revised 22 Aug. 2012; accepted 23 Sept. 2012; published online 3 Oct. 2012.
Recommended for acceptance by Y. Liu.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2012-01-0046.
Digital Object Identifier no. 10.1109/TPDS.2012.288.

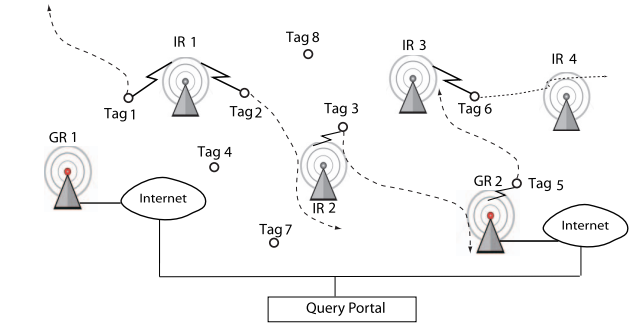
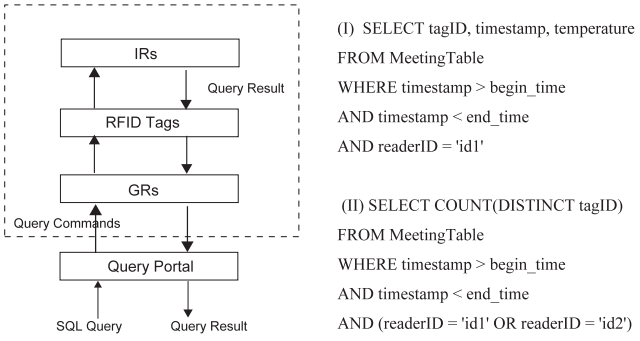


Fig. 1. An overview of FINDERS.



(a) Overall framework. (b) Query examples.

Fig. 2. Query distributed databases in FINDERS.

Sensing Platform (WISP) [10]) make measurements and write data into the tag's nonvolatile memory. Such on-tag sensor data, along with the tag ID, are subsequently acquired by a reader upon a future meeting event. In contrast to on-tag sensor data and meeting events that rely on the interaction between tags and readers and, thus, are often intermittent and unpredictable, the sensors colocated with a reader are battery-powered, generating on-reader sensor data periodically.

A collection of data discussed above intrinsically provide a discrete sampling of the movement of mobile nodes and their ambient environment. Although a FINDERS system consists of a small set of readers only and the short communication range further restricts opportunities to acquire meeting events and on-tag sensor data, such spatially sparse data samples are valuable for coarse-grained tracking and modeling in many applications. Given the extremely small storage space of tags, data in FINDERS are primarily hold by readers in their local flash memory. Each reader maintains a local database, and the readers all together constitute a distributed database system.

1.1.2 Data Communication

FINDERS aims to support applications in remote fields, where reliable communication infrastructures (such as cellular, WiMAX, and telemetry systems) are unavailable. Furthermore, due to sparse deployment in harsh wild environment with many obstacles, the readers are usually separated by a distance longer than the communication range of portable wireless technologies (e.g., Wifi or Zigbee). Thus, they cannot be connected via short-range radio; neither can they reliably access satellites. The field experimental setting also rules out licensed (such as UHF) radio. Only a handful of readers at convenient locations have stable network connections. They serve as gateways between FINDERS and conventional network infrastructure and are dubbed gateway readers, or GRs (see GRs 1-2 in Fig. 1). Meanwhile, most readers are isolated. The communication between such isolated readers, or IRs (e.g., IRs 1-4), is enabled by mobile tags that establish time-varying opportunistic links, forming an intermittently connected delay-tolerant network (DTN [11]) for data delivery.

Fig. 1 depicts examples of such distinctive communication paradigm for both upstream and downstream data transportation. First, to send data to an interested user, IR 1 writes them into Tag 2. When Tag 2 passes by IR 2, the former uploads its data to the latter. IR 2 subsequently writes the data into Tag 3, which has a trajectory through GR 2 and, thus, delivers the data to its destination via this gateway. Similarly, a user may transmit downstream data (such as commands or control packets) to IRs. Assume a command must be sent to IR 4. It can be written by GR 2 to Tag 5, which carries the data to IR 3. When Tag 6 passes by, it transports the data to IR 4.

Note that FINDERS is not proposed to replace conventional networks. Instead, it should be viewed as a supplemental way to transmit data when traditional communication networks cannot be established. As a matter of fact, if two readers can be connected by a wireless link, they should fully exploit such more reliable and efficient communication mechanism. Since the delay is extremely short in comparison with the transportation via mobile tags, the readers can be simply considered as a single (virtual) node in FINDERS.

1.2 Data Query in FINDERS

In this section, we introduce the problem of data query from a distributed database point of view, followed by discussions on the unique challenges and our solutions for achieving efficient data query in FINDERS.

1.2.1 Problem Description

FINDERS is a pervasive data acquisition and communication system, providing data to users with diverse needs. A straightforward scheme is to send all data to a server that is fully accessible by users. However, this naive approach which is commonly used in data acquisition systems over traditional networks is impractical, due to continuous updates on local databases and extremely limited communication capacity in FINDERS. In addition, the recent works on efficient RFID tag identification [12], [13], [14], [15], [16], missing tag detection [13], [17], [18], localization [19], [20], [21], [22], [23], and tag authentication and privacy [24], [25], [26], [27] are not applicable to the data query problem either. To improve the performance of query, data must be kept at the distributed databases of individual readers and fetched to users only if they are requested.

A query is sent to Query Portal (see Fig. 2a), which parses the request and instructs selected IRs to report the desired data. APIs are provided for users to query the databases via standard structured query language (SQL) statements. For example, the first query of (see Fig. 2b) requests to return the *tagID*, *meetingTime*, and *temperature* at

RFID reader "id1" during a specific time period. Another example is to check the unique number of tags that are seen by reader "id1" or "id2" during the time window. In general, a query can specify multiple time intervals and multiple sets of readers. Note that the requested time can be not only past or current but also future, which alerts readers to initiate query during a future interval. Moreover a request can be a "one-shot" relational query with a fixed answer set, or an ongoing continuous query that produces an unbounded stream of results. In addition to queries, a user may send other commands too, for example, to modify algorithmic parameters in data query and communication (such as reader scanning frequency), to request readers to report their status, or to delete obsolete database entries.

Note that although this work is related to [8] and both of them are based on the FINDERS architecture, it differs from [8] in several perspectives, calling for new solutions. First, [8] focuses on rostering, i.e., the creation of a list of unique IDs that appear in the network. It employs a dynamic space-efficient coding algorithm to construct data packets and relies on a unique packaging format tailored for rostering, to fit as much ID information as possible onto an RFID tag. On the other hand, this work introduces a general data query framework, aiming to gather not only tag IDs but also meeting events, on-tag sensor data, and on-reader sensor data. Since it does not assume that data include IDs only, the packaging format and dynamic space-efficient coding scheme introduced in [8] are no longer applicable here. Moreover, [8] largely deals with one-way communication, i.e., the IRs report IDs to the GR. There are control packets to be transmitted from GR to IRs, but they are all broadcasted. In this work, we consider two-way communications for transmitting commands, data, and feedback. A clustering algorithm is proposed to partition the network for efficiently utilizing transmission opportunities and reducing overhead.

1.2.2 Challenges

While data queries discussed above are similar to standard database operations, the transmission of query requests, results, and other control messages is nontrivial. The problem stems from the unique networking challenges in FINDERS, where communication can be established between a tag and a reader only, but not readers to readers or tags to tags. The communication must be initiated by a reader, which is in a sharp contrast to the symmetric transmission in most conventional networks. The tags serve as transportation vehicles, carrying data packets from one reader to another. However, given the short RFID communication range, the often uncontrolled tag mobility with great randomness, and the extremely limited storage space (ranging from tens to hundreds of bytes), the communication capacity of FINDERS is extraordinarily limited and the communication links are highly opportunistic, creating a sparse network where a tag is connected to a reader only occasionally. Whenever a communication opportunity is available, the reader must fully exploit the capacity of the tag. Since the tag's capacity is fixed, the challenge is to fit as much valuable information as possible onto the tag. Note that tags with different mobility patterns are suitable for carrying different packets. Moreover, redundancy is usually created during data transmission. Therefore, the reader must prioritize its data for efficient utilization of the precious communication opportunity. In general, the reader has many possible options to pack its data onto the tag. A distributed algorithm is indispensably needed to determine the set of data packets to be written onto the tag, to maximize its entropy, i.e., the effective information per bit transmitted, and consequently the overall system performance.

1.2.3 Contributions of This Work

The above challenges are addressed in this work to achieve efficient data query. We propose a distributed database query framework based on several communication and computing techniques specifically tailored for FINDERS. First of all, the RFID readers are grouped into clusters based on their intermittent connectivity and each tag is assigned a home cluster. When a communication opportunity becomes available between a reader and a tag, a dynamic appraisal is performed to determine the values of data packets according to information redundancy and tag mobility. A distributed algorithm based on 0-1 Knapsack model is devised to choose a set of packets, which together maximize their total (redundancy-excluded) value but do not exceed the capacity of the tag. We prototype the proposed data query system using Alien RFID gears and conduct experiments that involve 52 volunteers and last for 14 days to demonstrate its effectiveness. We also carry out simulations to evaluate the scalability of the proposed scheme under large-scale FINDERS systems.

The rest of the paper is organized as follows: Section 2 presents the query framework proposed for FINDERS. Section 3 elaborates implementation and testbed experiments. Section 4 illustrates simulation results. Finally, Section 5 concludes the paper.

2 Efficient Data Query in FINDERS

In this section we first present an overview of our proposed data query framework and then elaborate algorithmic details in support of effective query in FINDERS.

2.1 Overview of the Data Query Framework

The basic steps to execute a query in FINDERS are outlined below. First, the user submits the query that consists of standard SQL statements to Query Portal. Then, the Query Portal parses the SQL statements, creating one or multiple query commands. Subsequently, the query commands are transmitted to target IRs via intermediate tags and readers. Next, the target IRs respond the query and send corresponding data back to the Query Portal. Finally, the Query Portal combines received data and returns the complete SQL result to the user (see Fig. 2a).

Due to potentially long and divergent delays to acquire data from different IRs, the query must be asynchronous. The key challenge for achieving efficient data query in FINDERS is the extraordinarily low and intermittent communication opportunity and the extremely constrained capacity of RFID tags. Since data communication is so precious only possible when an RFID tag meets a reader, every communication opportunity should be efficiently

utilized to transmit the most valuable and deliverable data. To this end, two key issues must be addressed, which together determine the effectiveness and efficiency of data query in FINDERS.

First, given a communication opportunity, the reader needs to learn the capabilities of the tag to deliver data to desired destinations. In FINDERS, tags are vehicles for data transportation. Using a transportation station as an analogy, when a vehicle arrives at a terminal, one must know where the vehicle is heading, to determine the proper freight to load or passengers to board. Unfortunately, the tag's movement is nondeterministic, and thus, it is often impossible for the reader to learn the exact next hop(s) of the tag. However, with few exceptions in practical applications, the movement of an RFID tag follows some patterns, which can be modeled as nodal contact probabilities that have been widely discussed in the context of DTN [11]. For example, a tag may maintain a vector of contact probabilities to indicate its likelihood to meet individual readers in the future. Details of creating such a vector can be found in [28], [29]. However, this approach is impractical in FINDERS because a tag has such limit storage capacity that it cannot afford to carry a long vector of contact probabilities, one for each reader, with a total size proportional to the number of readers. To this end, we propose to compress the vector by grouping readers into clusters, where the communication opportunities among intracluster readers are high. Each tag is associated with a home cluster, to which it has the highest contact probability. Hence, a tag only carries a small, constant amount of information of its home cluster ID and corresponding contact probability. The details of clustering and home cluster association are to be elaborated in Section 2.2.

Second, given a communication opportunity, only a small amount of data can be written onto a tag. At the same time, a reader always holds a large volume of data in its local database, and therefore must prioritize its data to ascertain which data packets are most suited for the tag when a communication opportunity is available. As to be introduced in Section 2.3, we propose a dynamic appraisal scheme to estimate the values of data packets, and devise a distributed algorithm to maximize the total value of data loaded onto the tag. In a nutshell, our overall design principle is to shift most storage and computation load to readers, to save the precious tag space for maximized data transmission.

2.2 Clustering of RFID Readers and Tags

To create appropriate clusters, where the intracluster readers are well connected, the readers first learn and report their connectivity to the Query Portal, which in turn computes a flow-based link state matrix for cluster formation.

2.2.1 Link State Matrix

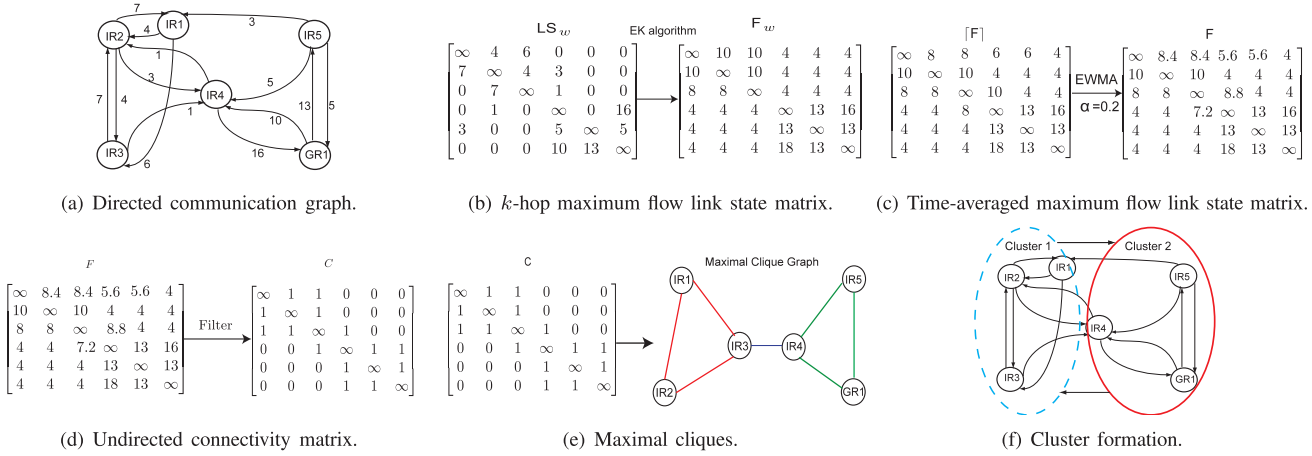
A tag always carries the ID of the reader that it meets most recently. Therefore, upon a meeting event, the reader can learn the previous stop of the tag, i.e., the upstream reader where the tag comes from. A reader maintains a list of upstream readers and estimates the communication capacity from each upstream reader to itself by counting the number of meeting events with tags coming from the upstream reader. For example, let l_{ij}^w denote such a count of tags that travel from Reader i (the upstream reader) to Reader j within a time window w. Note that l_{ij}^w is the *unidirectional* communication capacity from Reader i to Reader j, while l_{ji}^w (i.e., the capacity of the reverse link) is unknown to Reader j and can be dramatically different from l_{ij}^w. Similarly, Reader j learns the communication capacity from other upstream readers, forming a unidirectional link state vector $L_j^w = [l_{1j}^w, l_{2j}^w, \ldots, l_{nj}^w]$, where n is the total number of readers. Note that the actual size of L_j^w is far less than n, because only a handful of nearby readers can be the upstream readers of Reader j.

Each RFID reader periodically creates a *LinkState Packet* to report its unidirectional link state vector to the Query Portal. The LinkState Packet is small in size and generated once every time window only. Therefore, it introduces very limited overhead. The Query Portal combines LinkState packets (i.e., L_j^w) from all readers to construct the unidirectional link state matrix, $LS_w = [L_1^w, L_2^w, \ldots, L_n^w]$. Given the unique constraints (especially the intermittent connectivity) of FINDERS, it is essentially unavoidable that an IR may become unreachable during a time window. As a result, only partial (or even completely null) link state vector can be updated in this window. In this case, the Query Portal simply uses the best-known information (based on previous data received). Since time average values are used, the error due to temporary disconnection does not significantly affect the formation of clusters and the communication in FINDERS, as we observed in our experiments. The setting of time window should be determined by the requirement of the application and the mobility pattern of the tracking objects. If the time window is too small, there will be excessive overhead generated by propagating the LinkState packets, while if the time window is too large, it may fail to accurately reflect the mobility pattern of the tracking objects. In our experiment that involves human users (whose mobility naturally varies with a daily cycle), we set the time window to be one day.

The link state matrix obtained so far depicts a weighted, directional graph (see Fig. 3a, e.g.). It only reflects the direct communication capacity between readers. However, data delivery in FINDERS often involves indirect (multihop) transmissions. For example, a data packet can not only be transmitted by a tag from Reader i to Reader j directly, but also be delivered by different tags from Reader i to Reader x and subsequently from Reader x to Reader j. To capture such indirect communication capacity, the Query Portal creates a k-hop maximum flow link state matrix, denoted by F_w,

$$F_w = \begin{bmatrix} f_{11}^w & f_{12}^w & \cdots & f_{1n}^w \\ f_{21}^w & f_{22}^w & \cdots & f_{2n}^w \\ \cdots & \cdots & \cdots & \cdots \\ f_{n1}^w & f_{n2}^w & \cdots & f_{nn}^w \end{bmatrix}, \quad (1)$$

where f_{ij}^w is the maximum flow from Reader i to Reader j, under the constraint that any flow involves up to k hops of transmissions only. The maximum flow problem has been well studied. A modified Edmonds and Karp algorithm [30] with the k-hop constraint is adopted here to calculate f_{ij}^w, i.e., the k-hop maximum flow from Reader i to Reader j. Fig. 3b illustrates an example of F_w created according to LS_w.



(a) Directed communication graph. (b) k-hop maximum flow link state matrix. (c) Time-averaged maximum flow link state matrix.

(d) Undirected connectivity matrix. (e) Maximal cliques. (f) Cluster formation.

Fig. 3. An illustrative example of cluster formation.

F_w represents the communication capacity between readers estimated in the time window w. F_w may vary from a time window to another due to the unpredictable dynamics of tag mobility. Since F_w intrinsically serves as the routing metrics, it is highly desirable to be kept stable. To this end, we adopt Exponential Weighted Moving Average (EWMA) [7], [28] to create a time-averaged k-hop maximum flow link state matrix, F. When the Query Portal receives the first F_w, it sets $F = F_w$. Thereafter, EWMA is applied. More specifically,

$$F = \begin{cases} F_w & the\ first\ window \\ (1-\alpha) \times F + \alpha \times F_w & otherwise, \end{cases} \quad (2)$$

where α is constant between 0 and 1 to keep partial memory of historic status. Our experiments show that EWMA offers excellent stability and at the same time reacts effectively to small shifts. Moreover, it is simple and memory efficient, requiring only constant storage of historic data. Fig. 3c shows an example where multiple entries in F are updated by EWMA at the end of a time window.

2.2.2 Cluster Formation

The time-averaged k-hop maximum flow link state matrix F effectively describes the communication capacity between readers, based on which clusters are formed. A cluster includes a collection of RFID readers, among which any two readers have a mutual average maximum flow of at least ω. A simple clustering algorithm adopted in this work is outlined below. First, an undirected connectivity matrix C is defined according to F, where C_{ij} signifies if Readers i and j have a "strong" connection. More specifically, C_{ij} is determined as follows:

$$C_{ij} = \begin{cases} 0 & if\ F[i,j] < \omega \| F[j,i] < \omega, \\ 1 & otherwise. \end{cases} \quad (3)$$

An example of the undirected connectivity matrix C is illustrated in Fig. 3d. It represents a graph, where the vertices are readers and edges indicate strong connections between readers.

Second, a recursive backtracking procedure [31] is employed to identify a set of maximal cliques in the graph represented by C (see Fig. 3e, e.g.). A clique in the undirected graph is a subset of its vertices such that every two vertices in the subset are connected by an edge. A maximal clique is a clique that cannot be extended by including another vertex, i.e., a clique which does not exist exclusively within the vertex set of a larger clique. Therefore, the readers corresponding to the vertices included in a maximal clique all have strong connections, naturally forming a cluster, as illustrated in Fig. 3f. The communication capacity between two clusters is the total communication capacity between corresponding readers in the two clusters.

Once the Query Portal finalizes cluster formation, it encapsulates cluster IDs and members into the *ClusterInfo* packet and broadcasts it to all RFID readers such that each reader has a compressed global view of the network. Note that the ClusterInfo packet is created only when there is a change in cluster formation. Moreover, a progressive update scheme can be adopted to include only changes of clusters in the ClusterInfo packet, to reduce communication overhead.

2.2.3 Home Cluster Association

Each tag is associated with a *home cluster*, which consists of the reader it visits most frequently. To learn the frequency a tag visits a specific reader, the RFID reader records every meeting event with the tag in a time window to calculate their average meeting interval. Similar to (2), the meeting interval is time-averaged over windows by using EWMA. A tag carries its current home cluster ID and corresponding meeting interval T, which are initialized as an invalid ID and an arbitrary large value, respectively. Upon a meeting event, if the reader finds it has a shorter average meeting interval with the tag than the tag's current T, it overwrites the home cluster ID with its own cluster ID and updates T accordingly. Our experiments show that this distributed process quickly converges, yielding the correct home cluster association for tags, because a tag indeed visits its home cluster most frequently.

2.3 Dynamical Appraisal of Data Packets

The scarce and intermittent communication opportunity is the performance bottleneck for data query in FINDERS. With its extremely limited storage capacity, a tag obviously cannot carry all data at a reader upon a meeting event. Therefore, we propose a dynamic appraisal scheme to determine the values

of data packets, and adopt the 0-1 Knapsack algorithm to identify a set of most valuable packets without exceeding the tag's capacity for transmission.

The value of a data packet depends on the packet itself and the tag that is available for data transportation. First, bearing the nature of DTN where the transmission of a single copy packet is subject to high loss rate, data packets are often duplicated in FINDERS to achieve the desired delivery probability. For example, after a reader transmits a packet via a tag, it still keeps a copy the packet in its database, which can be transmitted again upon another communication opportunity if it is necessary. Therefore, multiple readers may have the same data packet in their databases, creating redundancy. The more the redundancy, the higher the chance that the packet can be delivered to its destination. Therefore, an individual copy of the packet depreciates its value when more redundancy is spread across the network. However, it is exorbitantly costly to keep tracking of such redundancy. In this work, we estimate the redundancy by two factors. First, from the global perspective, the longer a data packet has been propagated, the more redundancy is often created in the entire network. Second, each reader records the number of times it has transmitted a data packet, which serves as a local estimation of redundancy. Let u denote the value of the packet being appraised. We define $u = P(1-\eta)^{(t-t_o)}/m$, where P is a priority parameter (to differentiate different query tasks), η is a depreciation factor, t is the current time, t_o is the time that the data packet is generated, and m is the number of times the reader has transmitted the packet.

Besides the estimated redundancy, the value of a data packet also depends on the mobility of the tag. For example, a packet is more valuable (for the given tag) if the tag's home cluster includes the destination of the packet or is on the path to the destination. To this end, we introduce a parameter ζ, which is the fraction of the maximum flow from the source cluster to the destination cluster that passes through the tag's home cluster. ζ can be calculated by the maximum flow algorithm discussed earlier [30], but based on the graph of clusters (where each vertex is a cluster and the edge weight represents the communication capacity between two clusters). It indicates the likelihood that the tag transports the data packet in the right path toward its destination. Clearly, if the destination of the packet is in the tag's home cluster, $\zeta = 1$ because the entire maximum flow pours into the cluster. If a packet is to be delivered to multiple clusters, all of them are connected to a virtual destination node. Then, ζ is obtained similarly as discussed above. Consequently, $\zeta = 1$ for a broadcasting packet (that is destined to all clusters). Based on ζ, the value of the packet is adjusted to $u = \zeta P(1-\eta)^{(t-t_o)}/m$.

Upon a meeting event, the reader first appraises its data packets as discussed above, and then selects an optimal set of packets for transmission. A simple approach is to sort the packets according to their appraisals and greedily write the data packets with the highest values to the tag until there is no room available for the next packet. Another approach is to employ the 0-1 knapsack algorithm for more efficient utilization of the limited storage space of the tag. More specifically, the 0-1 Knapsack problem is formulated as follows:

$$\begin{aligned} Maximize: & \sum_{i=1}^{n} u_i x_i \\ Subject\ to: & \sum_{i=1}^{n} w_i x_i \le W,\ x_i \in \{0,1\}, \end{aligned} \tag{4}$$

where n is the total number of packets at the reader, W is the total storage space of the tag, and u_i and w_i are the appraisal and the size of Packet i, respectively. The 0-1 knapsack problem is NP-complete. A dynamic programming algorithm [32] is adopted here to determine 0-1 variables, $\{x_i \mid 1 \le i \le n\}$, i.e., the set of packets to be written into the tag, which together maximize the total value of the information being carried and at the same time do not exceed W. The solution for the knapsack problem is not our contribution. Most knapsack algorithms are applicable here. We simply adopts [32] in our implementation. It achieves close-to-optimal results with acceptable computation complexity. We have found it works well in our testbed. If an RFID reader does not support the needed computing power for such an algorithm, an approximate or heuristic approach [33] with less complexity can be adopted.

3 Prototype and Experiments

We have implemented a prototype to demonstrate distributed data query in FINDERS and to gain empirical insights into the design tradeoffs and practical considerations of the proposed query algorithm.

3.1 Implementation Issues

We have developed a prototype system by using Alien ALR-9900 readers and ALN-9540 tags (see Fig. 5, e.g.). The implementation closely follows the description in Section 2. It involves no modifications on the off-the-shelf tags or standard reader commands. Only a small amount of codes for link state dissemination and packet candidate creation, appraisal and selection are added to reader's program. Since the Alien RFID gear is Class1Gen2 (C1G2) standard compliant, our implementation can be generally applied to other standard passive RFID tags and readers as well.

We introduce five types of packets to support data query and transmit control information:

- *LinkState Packet*. A LinkState Packet is generated periodically by an IR or GR and transmitted to the Query Portal. A Linkstate packet contains the unidirectional link state vector observed by the corresponding reader (as discussed in Section 2.2).
- *ClusterInfo Packet*. A ClusterInfo packet is initiated by the Query Portal and broadcasted to the IRs and GRs. It contains the information about cluster formation and the connectivity among clusters. As introduced in Section 2.2, the ClusterInfo packet is created only when there is a change in cluster formation. A progressive update scheme is adopted to reduce communication overhead.
- *Query Packet*. A Query packet contains a query command issued by the Query Portal. A unique

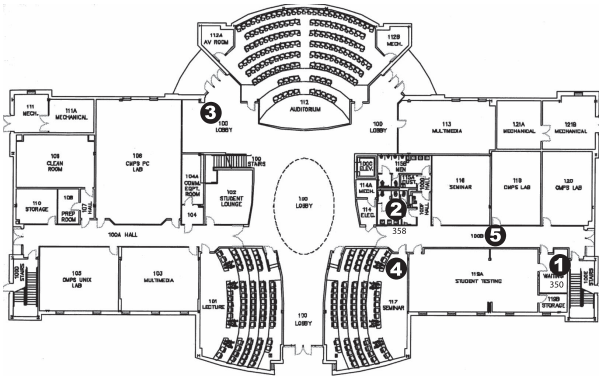


Fig. 4. Placement of RFID readers. Readers 1 and 2 are on the third floor, and Readers 3, 4, and 5 are on the first floor. Reader 1 serves as GR, while other readers are IRs.

sequence number (called Command ID) is associated with each command.

- *Reply Packet*. Zero to multiple *Reply* packets may be created by an IR, in response to a Query command. A *Reply* packet contains the result of the query.
- *Ack Packet*. The Query Portal periodically creates an *Ack* packet that contains a list of IDs of the commands which have been served. The *Ack* packet is broadcasted to IRs, facilitating them to eliminate unnecessary redundant copies of query commands and replies for efficient channel utilization.

In a nutshell, the *LinkState* and *ClusterInfo* packets are employed for cluster formation. *Query* packets are dispersed to the appropriate clusters that contain the query destinations or are on the paths toward to the destination clusters. *Reply* packets are created by IRs and delivered to the Query Portal. Meanwhile, *Ack* packets are distributed from the Query Portal to IRs to acknowledge the fulfillment of corresponding Query commands.

Each Alien ALN-9540-WR "Squiggle" tag has a total storage capacity of 128 bits available for applications. To expand tag capacity, a block-based scheme is adopted in FINDERS by leveraging the aggregated space of multiple passive tags, achieving a total capacity of tens to hundreds of bytes. The idea of constructing blocks has been presented in [7]. For the convenience of the readers, it is briefly outlined below. A block of tags is an integral unit attached to a mobile node. Each block includes a head tag and a number of storage tags. The head tag contains five control fields plus a number of packets. The control fields are Node ID, Tag ID, average interval, home cluster ID, and previous hop RFID reader ID, which consumes 10, 4, 10, 8, and 8 bits, respectively. Thus, the head tag has 88 bits left for packets. For each storage tag, besides the Node ID (10 bits) and Tag ID (4 bits), it has 114 bits for packets. A data packet includes four fields: a *Type* field of 3 bits, a *Query ID* field of 8 bits, a *Data* field with a variable length, and a packet-level *Time stamp* of 20 bits. The number of packets that can be carried by a tag depends on the available length of the *Data* field. It is worth mentioning that the tags in a block are read sequentially (but not simultaneously) by a reader. The tags within a block share the same Block ID. The Tag ID field of the head tag is predefined to 1111, and 1110 for the first storage tag, 1101 for the second storage tag, so on and so forth. To scan nearby nodes, the reader first sets its mask to 1111 for head tags only. Once a head tag is identified, the reader uses the Block ID and the predefined Tag ID to generate a unique mask for the next tag in the block, thus eliminating collision among storage tags in the same block. In our experiment, we employ a block of 3 tags or 4 tags, with a reading/writing delay of 185/170 or 245/240 ms, respectively. In the following discussion, we simply refer to a *tag* that is in fact a block of tags.

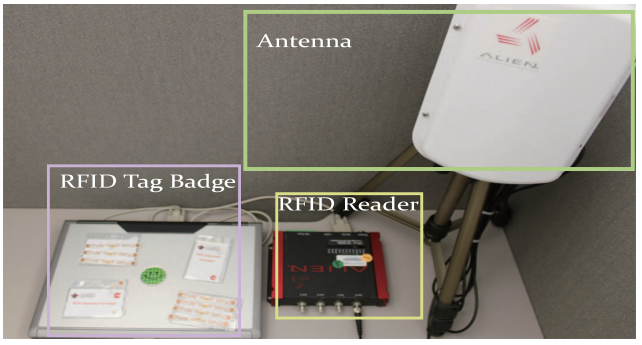


Fig. 5. Our Alien passive RFID gear.

Without suitable sensing elements at hand for integration with tags and readers, we focus on meeting events in our prototype. It requires only minor modification on the data collection program at the readers to incorporate other sensing data. Each GR or IR maintains a simple database that includes meeting events, active query commands it has received, query replies that it has learned so far, and the *LinkState* and *ClusterInfo* packets. Each database entry for a data packet contains the time the packet is generated, the time when the packet is received by the reader at the first time, the destination of the packet, and the destination cluster of the packet.

3.2 Testbed Setting

An experimental testbed has been set up to gain useful empiric insights of data query in FINDERS. Our testbed consists of five readers deployed in the Oliver Hall at the University of Louisiana at Lafayette that houses classrooms, research laboratories, and faculty offices for Computer Science and Engineering programs. Reader 3 (see Fig. 4) is deployed at the main entrance of the building on the first floor. Readers 4 and 5 are located at the entrance of two major classrooms (Room 116 and Room 117) on the first floor, respectively. Two readers (i.e., Readers 1 and 2 as shown in Fig. 4) are on the third floor, close to the doors of a small laboratory and a faculty office. Each reader is equipped with two side-by-side 6-dBi circular polarized antennae. Readers 2-5 serve as IRs, while Reader 1 is the GR, which connects to the Query Portal implemented on a workstation. All readers scan nearby tags at a frequency of once per second and the LinkState update window is set to 1 day.

Our experiment lasted 14 days, from 3/15/2011 to 3/28/2011. The first two days, i.e., from 3/14/2011 to 3/15/2011, are the initialization stage, where the system learns nodal contact probabilities and forms clusters. No data queries are generated during the initialization. Fifty two volunteers had participated in the experiment, including faculty members, senior Phd students (who spend more time in research labs

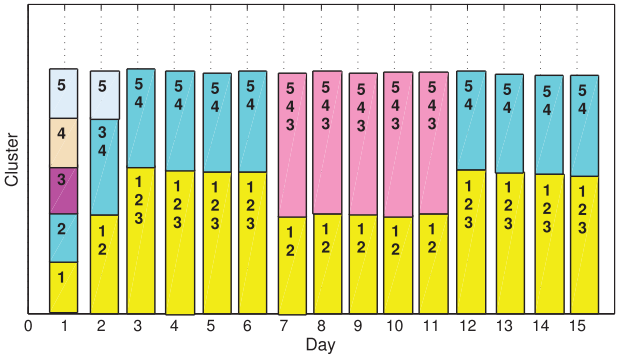


Fig. 6. Cluster formation. The colors indicate different clusters. The clusters become stable with minor updates only after the second day of our experiment.

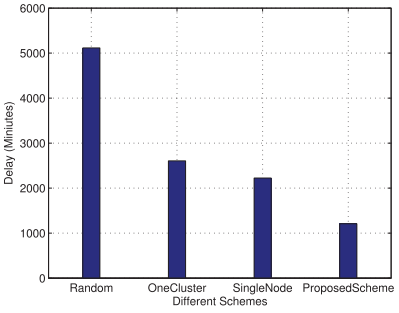


Fig. 8. Overall query delay.

than in classrooms), graduate students at M.S. level, and undergraduate students (who go to classrooms frequently and regularly). Some volunteers enter the building via its main entrance (where Reader 1 is deployed), while other do not. Each participant carries a badge, which is a typical conference badge with a block of tags enclosed. Among the 52 units, 32 of them contain three tags and the rest 20 units have four tags. Data queries are generated once every 2 hours with a random destination chosen among Reader 2 to Reader 5.

3.3 Experimental Results

We evaluate the performance of the proposed data query algorithm and compare it with three other schemes. Since it is impossible to execute multiple different algorithms simultaneously in an experiment, trace data are collected to run the competing schemes for fair comparison. "Random" is a naive approach where an IR randomly chooses a set of packets in its database that satisfy a query command for transmission. It often results in undesired high redundancy and long delay. In the "SingleNodeCluster" scheme, each individual IR or GR forms a single cluster. On the other hand, the "OneCluster" scheme puts all IRs and GRs in one cluster. "ProposedScheme" stands for the proposed scheme.

Appropriate clustering is crucial to the system performance. Thus, we first examine the cluster formation and its stability under practical application settings. Note that the cluster formation is an online process, subject to periodic update. Since nodes are mobile, the clusters are generally dynamic. Fig. 6 illustrates the convergence of the cluster formation process. With total five readers, they initially form five individual clusters. The colors indicate different clusters. They merge to three clusters on the second day and two clusters thereafter. In general, we observe that clustering converges quickly within two days and keeps stable with minor updates only after the second day of our experiment.

Figs. 7 and 8 show the percentage of query commands fulfilled and the average query delay (calculated based on the fulfilled queries only). The proposed algorithm achieves the best performance, significantly outperforming other schemes. The "OneCluster" scheme performs worse than "SingleNodeCluster," because the former is completely blind (without differentiation of destinations) and transmits all packets in the same way that is essentially broadcasting, thus resulting in inferior performance. "SingleNodeCluster" treats each node as a cluster, which may lead to high overhead in cluster maintenance. "Random" leads to the worst performance because it frequently makes wrong decisions to choose packets for transmission when there comes an communication opportunity.

Fig. 9 illustrates the delay distribution of fulfilled queries. Under "ProposedScheme," about 70 percent of the queries can be fulfilled within 1,000 minutes, significantly outperforming other schemes that can achieve about 10 percent at most. Moreover, nearly all queries can be satisfied in "ProposedScheme" within less than 4,000 minutes, while a large fraction of queries experience long delay under other schemes.

Fig. 10 presents a detailed look of data query results by illustrating the delay of each query during the entire experiment period. Queries generated during weekend always exhibit longer delay than the weekdays' queries. This is because few packets were able to be delivered on Saturday and Sunday due to low activity of students and faculty. Consequently, many packets had to stay in the IRs' queues during the whole weekend. Moreover, the accumulated data

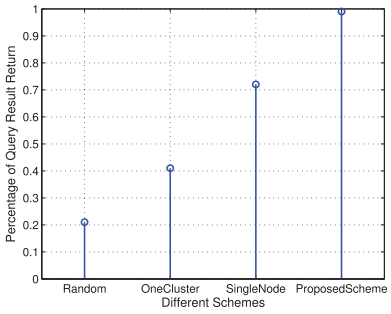


Fig. 7. Overall query success rate.

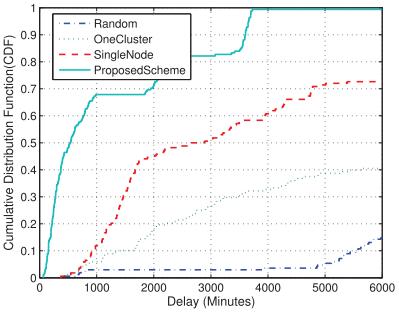


Fig. 9. Query delay distribution.

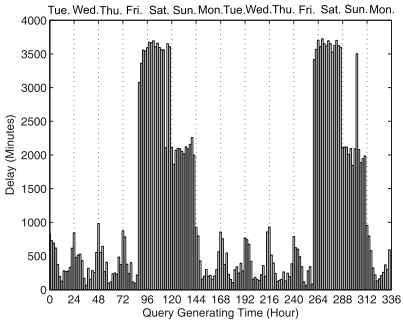


Fig. 10. Overall delay variation.

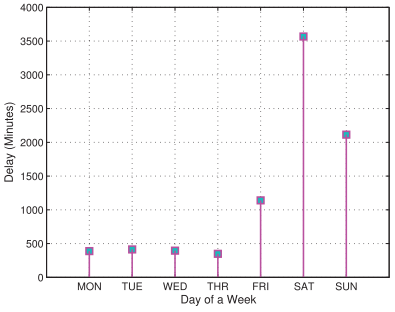


Fig. 11. Weekly delay variation.

result in further increased delay when they were transmitted on Monday. Similar pattern is observed in Fig. 11 that shows the average data query delay on each day of a week. The results closely match the activity pattern of students and faculty. During Monday through Thursday, the data query delay is low. The queries initiated at Friday afternoon experience a longer delay as the tag activity begins to decrease. The queries generated on Saturday and Sundays exhibit very long delay because no much data can be delivered during the weekends, and the longest delay is found for queries on Saturday because they have to wait for about two full days before been served. Fig. 12 further zooms in to show the delay of queries generated from the first to the 24th hour of a day. The first Tuesday of our experiment is chosen as an example, while similar results are observed on other weekdays as well. The query delay is low during daytime and high at night, which again shows the performance of query depends on the activity of mobile nodes who carry tags.

4 Simulation Results

Besides the experiments discussed above, extensive simulations are carried out to learn the performance trend of FINDERS under various settings with a large number of readers and tags, which are not practical to build in labs.

The simulated field is partitioned into 5×5 cells. A number of tags are distributed in the field and move according to power-law distribution, which is deemed as a realistic mobility model for delay-tolerant mobile networks [34]. More specifically, each mobile node has a home cell, which is randomly assigned in our simulations. Node i makes its decision to stay in the current cell or moves to one of the neighboring cells in every time slot. For example, if it is currently in Cell 0, it may move into one of four adjacent cells (e.g., Cells 1-4) or stay in Cell 0 in the next time slot. Its probability to be in Cell x is $P_i(x \mid 0) = P_i(x)/\sum_{z=0}^{4} P_i(z)$,

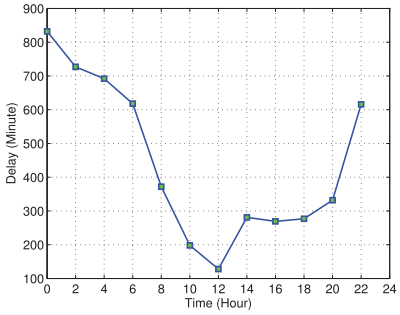


Fig. 12. Daily delay variation (Tue).

where $x = 0$ to 4 and $P_i(x) = (\frac{1}{d_i(x)})^{\beta}$. β is the exponent of the power-law distribution, which is set to 0.6 by default in our simulations. $d_i(x)$ denotes the distance from Cell x to Node i's home cell.

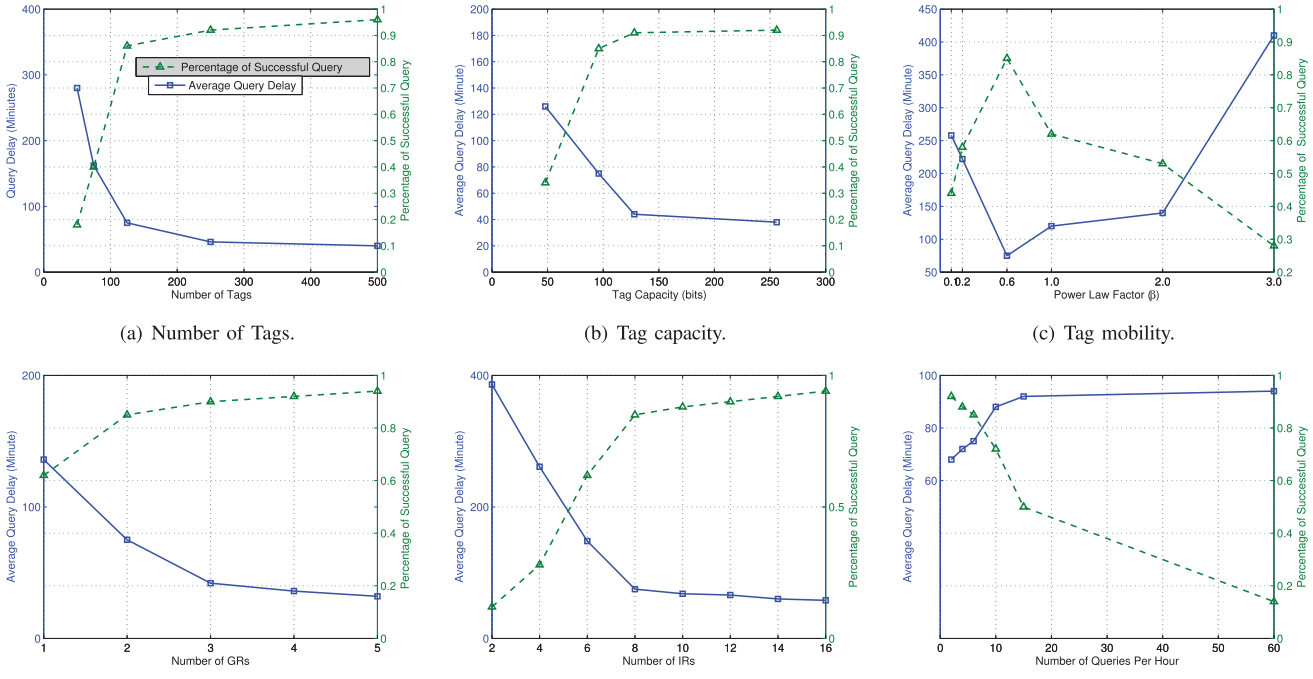
By default, 125 mobile nodes are randomly distributed in the field, each with a capacity of 96 bits. There are two GRs and eight IRs. The Query Command and Query Response are 16-bit and 48-bit long, respectively. The default query frequency is 6 queries per hour. We run simulations under the same configuration for 10 times and present the average results.

We first study the performance of data query by varying parameters related to tags. As shown in Fig. 13a, higher performance is achieved when the number of tags increases from 75 to 500, because more tags result in more communication opportunities (i.e., higher service rate). However, the gain becomes marginal when the tag density is higher than 200, because additional tags often carry unnecessarily duplicated queries and query results, useless for improving network performance. Similarly, increasing the capacity of tags (the main transportation vehicle in FINDERS) can improve the overall performance as shown in Fig. 13b. The power-law factor β is the decisive factor to the mobility pattern of tags. With a higher β, the tag has a higher chance to stay at its home location, which means less mobility. Therefore, the percentage of successful query is low. On the other hand, a very low β also has detrimental impact because it often results in a uniformly random mobility ineffective to guide packet delivery.

Fig. 13d illustrates the results with the number of GRs increasing from 1 to 5. It is obvious that deploying more GRs reduces query delay and improves successful query rate. Similarly, more IRs means a higher probability to deliver queries and query results. Consequently a higher query rate and lower query delay are observed in Fig. 13e when the number of IRs increases from 2 to 16. The frequency to initiate query commands is also an important factor. As can be seen from Fig. 13f, with the increase of query frequency, the percentage of successful query decreases. When the query frequency is above 20 per hour, the query delay becomes stable, because the network is saturated and the query delay is measured for successful queries only.

5 Conclusion

We have studied the problem of data query in intermittently connected passive RFID networks. To address the unique challenges in such an extremely resource-constrained network, we have propose a distributed data query framework that clusters RFID readers to reduce unnecessary data



(a) Number of Tags. (b) Tag capacity. (c) Tag mobility.

(d) Number of GRs. (e) Number of IRs. (f) Query frequency.

Fig. 13. Performance trend (simulation results).

transmission and establishes a 0-1 Knapsack model based on dynamic appraisal to choose the best set of packets, which together maximize their total (redundancy-excluded) value but do not exceed the capacity of a tag. We have carried out experiments by using Alien RFID gears that involve 52 volunteers for 14 days and performed large-scale simulations to evaluate the proposed data query framework.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under Grant CNS-0831823.

REFERENCES

[1] http://www.princeton.edu/mrm/zebranet.html, 2013.
[2] T. Small and Z.J. Haas, "The Shared Wireless Infostation Model - A New Ad Hoc Networking Paradigm (or Where There Is a Whale, There Is a Way)," *Proc. ACM MOBIHOC*, pp. 233-244, 2003.
[3] http://www.wu.ece.ufl.edu/projects/DeerNet/DeerNet.html, 2013.
[4] V. Dyo, S.A. Ellwood, D.W. Macdonald, A. Markham, C. Mascolo, B. Pasztor, S. Scellato, N. Trigoni, R. Wohlers, and K. Yousef, "Evolution and Sustainability of a Wildlife Monitoring Sensor Network," *Proc. Eighth ACM Conf. Embedded Networked Sensor Systems (Sensys)*, 2010.
[5] M. Wikelski, R.W. Kays, N.J. Kasdin, K. Thorup, J.A. Smith, and G.W. Swenson, "Going Wild: What a Global Small-Animal Tracking System Could Do for Experimental Biologists," *J. Experimental Biology*, vol. 210, pp. 181-186, 2007.
[6] Z. Yang and H. Wu, "Featherlight Information Network with Delay-Endurable RFID Support (FINDERS)," *Proc. IEEE Sixth Ann. IEEE Comm. Soc. Conf. Sensor, Mesh and Ad Hoc Comm. and Networks (SECON)*, pp. 55-63, 2009.
[7] Z. Yang and H. Wu, "FINDERS: A Featherlight Information Network with Delay-Endurable RFID Support," *IEEE/ACM Trans. Networking*, vol. 19, no. 4, pp. 961-974, Aug. 2011.
[8] Z. Yang and H. Wu, "Mobile Node Rostering in Intermittently Connected Passive RFID Networks," *Proc. IEEE Int'l Conf. Pervasive Computing and Comm. (PerCom)*, pp. 138-146, 2011.
[9] Y. Qiao, S. Chen, T. Li, and S. Chen, "Energy-Efficient Polling Protocols in RFID Systems," *Proc. ACM MOBIHOC*, pp. 267-275, 2011.
[10] D.J. Yeager, A.P. Sample, and J.R. Smith, "WISP: A Passively Powered UHF RFID Tag with Sensing and Computation," *RFID Handbook: Applications, Technology, Security, and Privacy*. CRC Press, 2008.
[11] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay Tolerant Network Architecture," draft-irtf-dtnrg-arch-02.txt, 2004.
[12] J. Myung and W. Lee, "Adaptive Splitting Protocols for RFID Tag Collision Arbitration," *Proc. ACM MOBIHOC*, pp. 202-213, 2006.
[13] T. Li, S. Chen, and Y. Ling, "Identifying the Missing Tags in a Large RFID System," *Proc. ACM MOBIHOC*, pp. 1-10, 2010.
[14] H. Vogt, "Efficient Object Identification with Passive RFID Tags," *Proc. Int'l Conf. Pervasive Computing*, 2002.
[15] V. Namboodiri and L. Gao, "Energy-Aware Tag Anti-Collision Protocols for RFID Systems," *Proc. IEEE Fifth Ann. Int'l Conf. Pervasive Computing and Comm. (PerCom)*, pp. 23-36, 2007.
[16] L. Xie, B. Sheng, C.C. Tan, H. Han, Q. Li, and D. Chen, "Efficient Tag Identification in Mobile RFID Systems," *Proc. IEEE INFOCOM*, pp. 1001-1009, 2010.
[17] W. Luo, S. Chen, T. Li, and S. Chen, "Efficient Missing Tag Detection in RFID Systems," *Proc. IEEE INFOCOM*, pp. 356-360, 2011.
[18] C.C. Tan, B. Sheng, and Q. Li, "How to Monitor for Missing RFID Tags," *Proc. 28th Int'l Conf. Distributed Computing Systems (ICDCS)*, 2008.
[19] L.M. Ni, Y. Liu, Y.C. Lau, and A.P. Patil, "LANDMARC: Indoor Location Sensing Using Active RFID," *Proc. IEEE First Int'l Conf. Pervasive Computing and Comm. (PerCom)*, pp. 407-415, 2003.
[20] D. Hahnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose, "Mapping and Localization with RFID Technology," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 1015-1020, 2004.
[21] K. Yamano, K. Tanaka, M. Hirayama, E. Kondo, Y. Kimuro, and M. Matsumoto, "Self-Localization of Mobile Robots with RFID System by Using Support Vector Machine," *Proc. IEEE/RSJ Int'l. Conf. Intelligent Robots and Systems*, pp. 3756-3761, 2004.
[22] Y. Zhao, Y. Liu, and L.M. Ni, "VIRE: Active RFID-Based Localization Using Virtual Reference Elimination," *Proc. Int'l Conf. Parallel Processing (ICPP)*, p. 56, 2007.

[23] C. Wang, H. Wu, and N.-F. Tzeng, "RFID-Based 3-D Positioning Schemes," *Proc. IEEE INFOCOM,* pp. 1235-1243, 2007.
[24] D. Henrici and P. Müller, "Providing Security and Privacy in RFID Systems Using Triggered Hash Chains," *Proc. IEEE Sixth Int'l Conf. Pervasive Computing and Comm. (PerCom),* pp. 50-59, 2008.
[25] T. Dimitriou, "A Secure and Efficient RFID Protocol That Could Make Big Brother (Partially) Obsolete," *Proc. IEEE Int'l Conf. Pervasive Computing and Comm. (PerCom),* pp. 269-275, 2006.
[26] L. Yang, J. Han, Y. Qi, C. Wang, T. Gu, and Y. Liu, "Season: Shelving Interference and Joint Identification in Large-Scale RFID Systems," *Proc. IEEE INFOCOM,* pp. 3092-3100, 2011.
[27] L. Yang, J. Han, Y. Qi, and Y. Liu, "Identification-Free Batch Authentication for RFID Tags," *Proc. IEEE 18th Int'l Conf. Network Protocols (ICNP),* pp. 154-163, 2010.
[28] Y. Wang and H. Wu, "DFT-MSN: The Delay Fault Tolerant Mobile Sensor Network for Pervasive Information Gathering," *Proc. IEEE INFOCOM,* pp. 1-12, 2006.
[29] Y. Wang and H. Wu, "Delay/Fault-Tolerant Mobile Sensor Network (DFT-MSN): A New Paradigm for Pervasive Information Gathering," *IEEE Trans. Mobile Computing,* vol. 6, no. 9, pp. 1021-1034, Sept. 2007.
[30] J. Edmonds and R.M. Karp, "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems," *J. ACM,* vol. 19, pp. 248-264, 1972.
[31] C. Bron and J. Kerbosch, "Algorithm 457: Finding All Cliques of An Undirected Graph," *Comm. ACM,* vol. 16, no. 9, pp. 575-577, 1973.
[32] *Introduction to Algorithms,* pp. 382-283. The MIT Press, 2001.
[33] S. Sahni, "Approximate Algorithms for the 0/1 Knapsack Problem," *J. ACM,* vol. 22, no. 1, pp. 115-124, 1975.
[34] J. Leguay, T. Friedman, and V. Conan, "DTN Routing in a Mobility Pattern Space," *Proc. ACM SIGCOMM '05,* pp. 276-283, 2005.

Zhipeng Yang (S'07) received the BS and MS degrees in computer science from Tianjin University, Tianjin, China, in 2001 and 2004, respectively, and has been working toward the Phd degree in computer science at The Center for Advanced Computer Studies, University of Louisiana, Lafayette, since 2007. From 2004 to 2006, he was a software engineer in Nortel and Lucent China. His current research interests include delay-tolerant networks, radio frequency identification (RFID) systems, and wireless sensor networks. He is a student member of the IEEE.

Ting Ning (S'10) received the BS and MS degrees in computer science from Lanzhou University, Lanzhou, China, in 2005 and 2008, respectively, and has been working toward the PhD degree in computer science at The Center for Advanced Computer Studies, University of Louisiana, Lafayette, since 2008. Her current research interests include delay-tolerant networks, the application of game theory, and radio frequency identification (RFID) systems. She is a student member of the IEEE.

Hongyi Wu (M'02) received the BS degree in scientific instruments from Zhejiang University, Hangzhou, China, in 1996, and the MS degree in electrical engineering and the PhD degree in computer science from the State University of New York (SUNY) at Buffalo in 2000 and 2002, respectively. Since then, he has been with the Center for Advanced Computer Studies (CACS), University of Louisiana at Lafayette (UL Lafayette), where he is currently a professor and hold the Alfred and Helen Lamson Endowed professorship in Computer Science. His research spans delay-tolerant networks, radio frequency identification (RFID) systems, wireless sensor networks, and integrated heterogeneous wireless systems. He received the US National Science Foundation (NSF) CAREER Award in 2004 and UL Lafayette Distinguished Professor Award in 2011. He is a member of the IEEE.