Efficient Data Query in Intermittently-Connected Mobile Ad Hoc Social Networks

Yang Liu, *Student Member, IEEE*, Yanyan Han, Zhipeng Yang, *Student Member, IEEE*, and Hongyi Wu, *Member, IEEE*

Abstract—This work addresses the problem of how to enable efficient data query in a Mobile Ad-hoc SOcial Network (MASON), formed by mobile users who share similar interests and connect with one another by exploiting Bluetooth and/or WiFi connections. The data query in MASONs faces several unique challenges including opportunistic link connectivity, autonomous computing and storage, and unknown or inaccurate data providers. Our goal is to determine an optimal transmission strategy that supports the desired query rate within a delay budget and at the same time minimizes the total communication cost. To this end, we propose a centralized optimization model that offers useful theoretic insights and develop a distributed data query protocol for practical applications. To demonstrate the feasibility and efficiency of the proposed scheme and to gain useful empirical insights, we carry out a testbed experiment by using 25 off-the-shelf Dell Streak tablets for a period of 15 days. Moreover, extensive simulations are carried out to learn the performance trend under various network settings, which are not practical to build and evaluate in laboratories.

Index Terms—Data query, mobile ad hoc social networks, centralized optimization model, distributed protocol, testbed experiment, simulations

1 Introduction

Social networking is among the fastest growing information technologies, as evidenced by the popularity of such online social network sites as Facebook, Twitter, LinkedIn and Google+ that continue to experience explosive growth.

In contrast to the popular web-based online social networks that rely on the Internet infrastructure (including cellular systems) for communication, this paper focuses on Mobile Ad-hoc SOcial Network (MASON), an autonomous social network formed by mobile users who share similar interests and connect with one another by exploiting the Bluetooth and/or WiFi connections of their mobile phones or portable tablets. A MASON is often created for a local community where the participants have frequent interactions, e.g., people living in an urban neighborhood, students studying in a college, or tourists visiting an archaeological site. Its size varies from a large group (for instance, all the students in a university) to a small cluster (such as members of a school band). It may serve a community over a long span of years, or be temporary to last for as short as a few hours only (e.g., for social networking among a group of tourists).

1.1 System Overview

An individual MASON is incomparable with online social networks in terms of the population of participants, the number of social connections and the amount of social media. However MASONs gain significant value by serving as a supplement and augment to online social networks and by effectively supporting local community-based ad-hoc social networking. For example, it helps discover and update social links that are not captured by online social networks and allows a user to query localized data such as local knowledge, contacts and expertise, surrounding news and photos, or other information that people usually cannot or do not bother to report to online websites but may temporarily keep on their portable devices or generate upon a request.

This work addresses the problem of how to enable efficient data query in MASONs. Consider a MASON with N nodes. Each node can be a query issuer or a data provider, or more commonly act in both roles for different query requests. The queries fall into C categories. Each node has certain expertise to answer a query. Let E denote the expertise matrix, where E_i^c indicates the expertise of Node i to answer a query in Category c, i.e., the probability that Node i can provide a satisfactory answer to a query in Category c. A query is created by a query issuer. It is delivered by the network toward the nodes that can successfully provide an answer (i.e., data providers). If a data provider receives the query, it sends the query reply to the query issuer.

Our goal is to determine an optimal transmission strategy that supports the desired query rate and at the same time minimizes the total communication cost. The formal problem formulation will be given in Section 2.1.

1.2 Unique Challenges

The use of free, short-range radio is highly desired for a diversity of MASON applications. At the same time, however, it results in a distinctive communication paradigm characterized by intermittent link connectivity and

• *The authors are with the Center for Advanced Computer Studies, University of Louisiana at Lafayette, Lafayette, LA70504. E-mail: {yxl0782, wu}@cacs.louisiana.edu, yanyan.ull@gmail.com, zhipengyang@hotmail.com.*

Manuscript received 23 Jan. 2014; revised 21 Apr. 2014; accepted 23 Apr. 2014. Date of publication 28 Apr. 2014; date of current version 8 Apr. 2015. Recommended for acceptance by J. Lloret.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPDS.2014.2320922

autonomous computing and storage. More specifically, the data query in MASONs faces the following unique challenges.

(1) Opportunistic link connectivity. The connectivity of MASONs is very low and intermittent, forming a sparse network where a node is connected to other nodes only occasionally. This is in a sharp contrast to online social networks, where users always have reliable Internet connections. The data delivery delay in MASONs is potentially long, due to the loose connectivity among nodes. Fortunately, such delay, thought not desirable, is usually tolerable by many data query applications in MASONs that are often delay insensitive.

(2) Autonomous computing and storage. Central servers are employed to store and process user data in online social networks. Such servers are, however, no longer available in MASONs, where individual portable devices must perform distributed data storage and computation. It is well known that portable devices have limited computing, storage and energy capacity. Nevertheless, such constraints are particularly disadvantageous to MASONs, because a node must process data in a distributed manner and store them locally for a much longer time before sending them to another node, due to intermittent connectivity.

(3) Unknown or inaccurate expertise. When a node issues a query, it is often unaware of the nodes that have sufficient expertise to answer the query, since the cost is prohibitively high to construct a structure to index data and data providers like P2P networks. It is obviously inefficient either to frequently flood queries, which are expensive and often considered spams. Worse yet, in practice, a mobile node hardly knows its probability to answer queries in each category precisely. It may initially claim its expertise based on the mobile user's social roles and available resources. But such initially claimed expertise is often inaccurate.

The above characteristics make the data query in MASONs a very unique, interesting, and challenging problem, rendering not only conventional data query schemes for well-connected computer systems but also distributed solutions for mobile ad hoc networks [1], [2], [3], [4], [5], [6] and mobile (online) social networks [7] inapplicable here. Only a handful of works have considered data query in opportunistic network settings. For example, Osmosis [8] employs an epidemic approach to perform file lookup in pocket switched networks. While it is simple and reliable, the communication overhead is very high due to the flooding-like epidemic routing. DelQue [9] aims to query geolocation-based information. It assumes each node moves according to a given schedule and adopts a semi-Markov model to predict nodal meeting events, in order to identify a proper relay to carry the query to the target location and bring the interested information back to the source. Yang et al. [10] proposes a distributed database query framework based on several communication and computing techniques specifically tailored for RFID networks. Neither of them efficiently support data queries in MASONs. On the other hand, although several routing algorithms have been proposed for opportunistic networks by exploiting social relations among mobile users to achieve efficient routing [11], [12], [13], [14], [15], [16], they are developed for generic communications, without consideration of the unique needs and constraints in data query. Among them, Zhu et al. [16] is the most recent one, where exploits a distributed community partitioning algorithm to divide a DTN into smaller communities. For intra-community communication, a utility function convoluting social similarity and social centrality with a decay factor is used to choose relay nodes. For inter-community communication, the nodes moving frequently across communities are chosen as relays to carry data to destination efficiently. Although Zhu et al. [16] introduces a solution for DTNs which leverages social properties and mobility characteristics of users, it is not truly applicable for the data query in MASONs, because it does not capture the inherent features for the query delivery in MASONs, hence the nodes are not helpful for each other by making the correct decisions to carry queries to satisfactory nodes.

1.3 Our Contribution

In this work, we first formulate the optimization problem for the data query in MASONs. Then, we develop a state-diagram-based analytic model to derive the communication overhead and query rate as the functions of transmission strategy. A branch and bound algorithm is adopted to discover the optimal transmission strategy that minimizes the total communication cost while achieving a target query rate.

The optimization model is centralized, thus unpractical for real world implementation. However, it offers useful insights for the development of a distributed data query protocol. The proposed protocol is based on two key techniques. First, as motivated by the analysis, it employs "reachable expertise" as the routing metric to guide the transmission of query requests. Second, it exploits the redundancy in query transmission, which can effectively improve the query delivery rate in practice if it is properly controlled.

To demonstrate the feasibility and efficiency of the proposed data query protocol and to gain useful empirical insights, we have carried out a testbed experiment using off-the-shelf Dell Streak tablets. Our experiment involves 25 volunteers and lasts for 15 days. Moreover, extensive simulations (based on codes extracted from our prototype implementation) are carried out to learn the performance trend under various network settings, which are not practical to build and evaluate in laboratories.

The remainder of this paper is organized as follows. Section 2 introduces the proposed data query scheme, including a theoretic optimization model and a practical protocol. Section 3 presents a testbed experiment and results. Section 4 discusses large-scale simulations under real-world mobility traces and power-law mobility model. Finally, Section 5 concludes the paper.

2 Proposed Data Query Scheme

While MASONs offer interesting opportunities to support ad hoc data query, its capacity is unsurprisingly low compared to many other data networks due to its extremely limited and nondeterministic communication opportunities. To learn the essence of optimal query delivery and to understand the performance upper bound, we first carry out a preliminary analytic study of the data query in MASONs before moving into the detailed protocol design and

evaluation. Based on the insights gained from our analysis, a distributed data query protocol is proposed, aiming to enable highly efficient ad hoc query under practical MASON settings.

2.1 Preliminary Analysis

In this section, we first formulate the problem, and then introduce an analytic model to obtain an optimal solution, followed by discussions on numeric results and insights for developing a practical distributed data query protocol.

2.1.1 Assumptions and Problem Formulation

The communication in MASONs depends on nodal meeting events. The time interval between two consecutive meeting events between two nodes, e.g., Nodes i and j, is denoted by a random variable T_{ij}. We consider generic nodal mobility in this work without assuming any specific distribution of T_{ij}.[1] For analytic tractability, we make the following two assumptions in data transmission.

Assumption 1. *Given the intermittent network setting, we assume that the communication capacity is ruled by nodal meeting opportunities.*

In other words, we assume the communication delay is dominated by nodal meeting intervals. When two nodes meet, the channel bandwidth is sufficient for them to exchange data packets with negligible delay.

Assumption 2. *Multiple copies of a query request may exist in the network, but we assume a node receives and forwards the same request only once.*

For a query request, we define a binary transmission matrix, X, where $X_{ij} = 1$ indicates that Node i sends the request to Node j, if the former holds the request when it meets the latter. If $X_{ij} = 0$, then Node i does not send the request to Node j even if it carries the request upon meeting Node j. Note that X_{ij} and X_{ji} can be different.

The transmission of query reply follows standard one-to-one routing with known source and destination. Its expected transmission delay has been well studied [11], [12], [13], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37]. Therefore we focus on the delivery of query requests only in this analysis (but a complete protocol is designed and implemented as to be discussed in later sections). Our goal is to determine the optimal X such that the total communication cost, defined as $C(X)$, is minimized, while at the same time the probability to deliver the query to a data provider within a given delay budget δ (denoted as $P(X)$) reaches the desired threshold, β. More specifically, the problem is formulated as follows:

$$\begin{aligned} Objective: &\quad min(C(X)), \\ S.t.: &\quad P(X) \geq \beta. \end{aligned} \tag{1}$$

Under the assumption of sufficient channel bandwidth, different query requests do not contend for communication resource and thus can be treated independently. To this end, we next simply analyze the delivery of a single query request to derive $C(X)$ and $P(X)$, and to optimize X.

1. It is out the scope of this research to model user mobilities. The readers are referred to the literatures [17], [18], [19] that have reported extensive studies on mobility modeling.

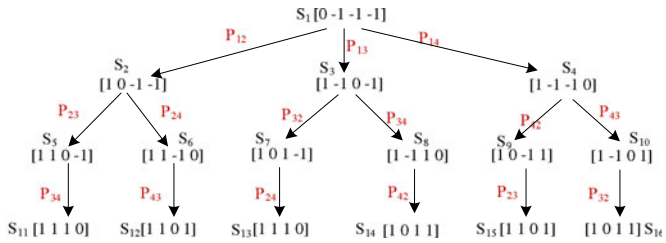


Fig. 1. The state diagram shows all possible transmissions of a query in a network with four nodes. Without loss of generality, we let Node 1 be the query issuer. Since we focus on a single query, the category of the query (i.e., c) is not shown in the state diagram.

2.1.2 Analytic and Optimization Model

Our analysis is based on a state diagram. Each state is a vector with N elements, i.e., $S = [s_1, s_2, \ldots, s_N]$, where $s_i = -1$ signifies Node i has not received the query request, $s_i = 0$ indicates Node i is carrying the query request but has never transmitted it to another node, and $s_i = 1$ means Node i has received the query request and forwarded it once. Fig. 1 illustrates an example of the state diagram for a network with four nodes. Since we focus on a single query in this analysis, the category of the query (i.e., c) is not shown in the diagram.

Without loss of generality, we let Node 1 be the query issuer. Thus the initial state is $S = [0, -1, -1, \ldots, -1]$ (see Fig. 1). The state transits, as depicted by an arrow in the diagram, when the query request is transmitted from one node to another, e.g., from Node i to Node j. Such a state transition is denoted by L_{ij}. Since we have assumed that a node transmits and receives the same request only once, L_{ij} is possible only if $s_i = 0$ and $s_j = -1$, i.e., Node i is carrying the query request but has never transmitted it while Node j has not received the query request. Note that this is the necessary condition to enable a transmission, but Node i does not have to transmit the query to Node j if such a transmission is deemed inefficient. As a matter of fact, it is the goal of our analysis and optimization to determine the most efficient transmissions.

The state diagram (including the states and transitions) forms a tree structure. The initial state is the root of the tree. As shown in Fig. 1, a state may appear multiple times due to different transition paths to reach it. We treat such identical states separately (as if they are different states).

A state can be in three status, i.e., *active*, *negatively terminated*, or *positively terminated*, as defined below.

Definition 1. *An active state is a state that allows further transitions.*

The query request will be further transmitted under an active state. An active state must have a 0 element, i.e., $s_i = 0$, which is also called an active element.

Definition 2. *Node i is an* active element *of an active state if and only if $s_i = 0$.*

Fig. 1 depicts active states only. Each active state may remain active or be terminated in two possible ways, resulting in a negatively terminated state or a positively terminated state.

Definition 3. *An active state becomes a* positively terminated state *if the query request is answered by the active element.*

Upon receiving a query request in Category c, Node i has a probability of E_i^c to answer the query. If the query is answered within its delay budget, it will no longer be forwarded it to another node. This essentially terminates the query process given the assumption that a node forwards the same request only once. Otherwise, with a probability of $1 - E_i^c$, Node i may remain active and carry the request that is ready to be transmitted to another node.

Definition 4. *An active state becomes a* negatively terminated state *if the query request is dropped by the active element.*

A query request is dropped if its delay budget expires. As a result, the state includes no active element and the query process is terminated without a successful reply.

Fig. 1 depicts all possible transmissions of a query in a network. Our goal is not to execute all such transmissions which lead to high communication overhead, but instead to perform selective transmissions to minimize the communication cost. To this end, we have introduced the transmission matrix, X, where $X_{ij} = 1$ signifies that Node i will send the request to Node j, when the former meets the latter with $s_i = 0$ and $s_j = -1$; or the transmission will not be performed otherwise.

With X as the variable to be optimized, we now analyze the probability to reach each state. Since the state diagram forms a tree structure, there is a unique path from the root (i.e., the initial state) to a given state $S = [s_1, s_2, \ldots, s_N]$. Once again, identical states may appear in the tree and they are treated as separate states. Let L_S denote the path from the root to S, which consists of a sequence of transmissions $\{L_{U_1^S U_2^S}, L_{U_2^S U_3^S}, \ldots, L_{U_{K-1}^S U_K^S}\}$, where $\{U_1^S, U_2^S, \ldots, U_K^S\}$ are the set of nodes involved in the transmissions in sequence. For example, the path from the root to State $S_{14}[1, 0, 1, 1]$ includes such links as L_{13}, L_{34}, and L_{42}. Each link introduces a transmission delay. The total delay to reach the state is $\sum_{i=1}^{K-1} T_{U_i^S U_{i+1}^S}$.

We are interested in positively terminated states. A state S can be positively terminated if and only if the following three conditions are satisfied. First, the transmission matrix is configured such that $X_{U_i^S U_{i+1}^S} = 1$, $\forall 1 \le i \le K-1$, forming a valid path from the root to State S. Second, the total delay to reach the state is no greater than the delay budget δ, i.e., $\sum_{i=1}^{K-1} T_{U_i^S U_{i+1}^S} \le \delta$. Note that δ can be very large under practical settings. As a matter of fact, MASON users are often not anxious to receive prompt query replies and thus may tolerate long delay. In this case, the delay distributions in the above formula can be reduced to simple probabilities as to be discussed in the next section. Third, the last node on the path (i.e., Node U_K^S) can answer the query while others along the path (i.e., Nodes U_i^S, $\forall 1 \le i \le K-1$) cannot. Therefore the probability that State S is positively terminated is

$$P_S(X) = \prod_{i=1}^{K-1} X_{U_i^S U_{i+1}^S} \times E^c_{U_K^S} \prod_{i=1}^{K-1} \left(1 - E^c_{U_i^S}\right) \times Pr\left\{ \sum_{i=1}^{K-1} T_{U_i^S U_{i+1}^S} \le \delta \right\}. \quad (2)$$

Since the states are uncorrelated, the total probability to reach a positively terminated state, i.e., the probability to deliver the query request to a node that can answer the query is $P(X) = \sum_S P_S(X)$. With proper manipulation, we arrive at the following formula:

$$P(X) = \sum_{i=1}^{N-1} \sum_{k_1 \ne k_0}^{N} \sum_{k_2 \ne k_0, k_1}^{N} \cdots \sum_{k_i \ne k_0, k_2, \ldots, k_{i-1}}^{N} \prod_{j=0}^{i-1} X_{k_j k_{j+1}} \times E^c_{k_i} \prod_{j=0}^{i-1} \left(1 - E^c_{k_j k_{j+1}}\right) \times Pr\left\{ \sum_{j=0}^{i-1} T_{k_j k_{j+1}} \le \delta \right\}. \quad (3)$$

Note that, the total number of add operations in the above equation equals the number of states, which we have found to be $N_S = \sum_{i=1}^{N-1} A_{N-1}^i$, where A_{N-1}^i is the permutation of i out of $N-1$ elements, i.e., $A_{N-1}^i = i!/(N-1-i)!$

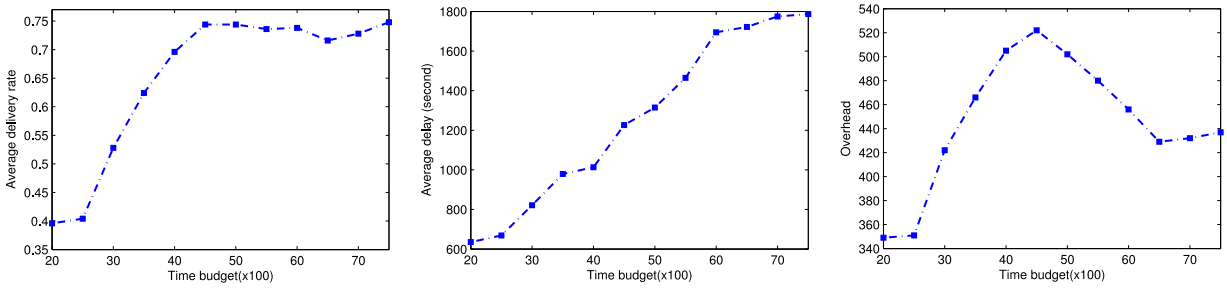
The communication cost in a wireless network is often proportional to the number of transmissions. The more the transmissions, the higher the consumption of energy and storage space. It is out the scope of this work to define the best cost function. We simply let $C(X)$ be the total number of transmissions involved in the delivery of a query request. Let $D(S)$ denote the depth of State S in the tree. Obviously, $D(S)$ represents the number of transmissions (i.e., the cost) needed to transit from the initial state to S. Note that, the cost of $D(S)$ is incurred as long as State S is reached, no matter whether the query can be answered or not. Thus $C(X)$ is given below:

$$C(X) = \sum_S D(S) \frac{P_S(X)}{E^c_{U_K^S}}. \quad (4)$$

$P(X)$ and $C(X)$ are obtained via Eqs. (3) and (4), and then plugged into Eq. (1). A branch and bound algorithm [38] is adopted here to discover the optimal transmission matrix X, in order to minimize $C(X)$ while ensuring $P(X) \ge \beta$.

2.1.3 Numeric Results and Insights

Fig. 2 shows the numeric results of the optimization model based on Haggle trace [39] (to be detailed in Section 1). β is set to be 0.5. Under very small δ, many queries cannot be delivered within the delay budget, resulting in low delivery rate. When $\delta < 30$, the delivery rate is in fact lower than the expected threshold (i.e., β). By following the optimization model, the nodes do not make unnecessary attempts to transmit them, and accordingly the overhead is low. With a longer delay budget, more queries can reach the corresponding data providers and thus the delivery rate naturally increases. At the same time, delay and overhead increase too because more transmissions with longer delay are aggressively attempted. However, when δ is sufficiently large, many options of routing paths become available (that all satisfy the delay budget), allowing the optimization model to choose the one with the lowest overhead (i.e., the one that involves the least transmissions). This explains why overhead decreases under large δ. Such a choice often sacrifices delay, as long as it does not exceed the allowed budget. Therefore, the average delay monotonically increases with δ.



(a) Delivery rate. (b) Delivery delay. (c) Overhead.

Fig. 2. Numeric results of optimization.

While the above optimization model can be implemented by each individual node, it is intrinsically centralized (requiring global information), and thus unpractical for real world implementation. However, it offers useful insights for the development of a distributed data query protocol. In particular, the essence of the optimization is to first learn the probabilities to deliver the query along different paths to different nodes, and then decide the optimal paths by striking the balance between cost and delivery probability. This insight stimulates us to develop a distributed scheme to learn "reachable expertise", which can effectively guide the transmission of a query to the node(s) with sufficient expertise to answer it.

2.2 Protocol Design

In this section, we introduce a distributed protocol for the data query in MASONs. It is based on two key techniques. First, as motivated by our analytic and optimization model discussed above, it employs "reachable expertise" as the routing metric to guide the transmission of query requests. Second, it exploits the redundancy in query transmission. Redundancy is not considered in the analysis due to its intractability, but can effectively improve the query delivery rate in practice if it is properly controlled.

2.2.1 Routing Metric

The delivery of query depends on a routing metric, which is updated routinely and maintained separately from the routing algorithm itself. We first introduce such a metric, i.e., reachable expertise, that guides query transmission.

As introduced in Section 2.1.2, each node has certain expertise to answer a query. We let E_i^c denote the expertise of Node i to answer a query in Category c. We have assumed E_i^c is known in our analysis. In practice, however, it is non-trivial to properly define the expertise, because a mobile node hardly knows precisely its probability to answer queries in each category. It may initially claim its expertise based on the mobile user's social roles (e.g., professions), interests, and available resources. But such initially claimed expertise is often inaccurate. Therefore, after initialization, the expertise should be updated according to the feedbacks from other nodes, especially the query issuers.

In this research, we adopt the exponentially weighted moving average (EWMA) to maintain and update expertise. More specifically, we have

$$E_i^c \leftarrow (1-\mu)\left[E_i^c\right] + \mu F_i^c, \tag{5}$$

where $0 \le \mu \le 1$ is a constant weight to keep partial memory of historic status, $[E_i^c]$ is the expertise before it is updated, and F_i^c is the feedback score for queries that Node i has answered in Category c. Various feedback rating schemes can be adopted to determine F_i^c. In this research, we employ a simple scheme, which supports quick convergence as to be discussed in Section 3 and shown in Fig. 3h.

The expertise indicates the capability of a node to answer queries, but itself is insufficient to guide query transmission. For example, a node may have high expertise, but is not reachable by the query issuer and thus becomes less helpful to answer the query. To this end, we define k-hop reachable expertise. As discussed earlier, the MASON users can often tolerate long delay. Thus, the delay random variables in Eq. (3), i.e., $T_{U_i^S U_{i+1}^S}$, may be reduced to simple nodal contact probabilities. Let p_{ij} denote the probability that Nodes i and j meet. The maintenance and update of p_{ij} have been discussed extensively in DTN networks [20], [21], [24], [25]. The k-hop reachable expertise is calculated as follows:

$$E_i^c(k) = 1 - \prod_{j\in\Phi}\left(1 - p_{ij}\cdot E_j^c(k-1)\right), \tag{6}$$

where Φ is the set of nodes that Node i meets frequently. $E_i^c(k)$ indicates the probability that Node i can deliver the query within k hops to a node that can answer the query. Clearly, $E_i^c(0) = E_i^c$ and $E_i^c(1) = 1 - \prod_{j\in\Phi}(1 - p_{ij}\cdot E_j^c)$. Node i collects $\{E_j^c(k-1) | \forall\ j \in \Phi\ and\ 0 < k \le N\}$ whenever it meets other nodes, and periodically makes an update on $E_i^c(k)$ according to Eq. (6).

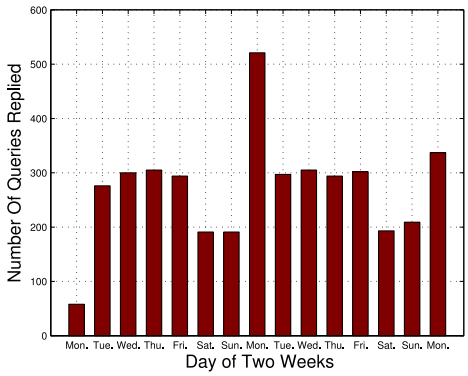
Based on the k-hop reachable expertise, we define the aggregated reachable expertise,

$$AE_i^c = 1 - \prod_{k=1}^{N}\left(1 - E_i^c(k)\right), \tag{7}$$

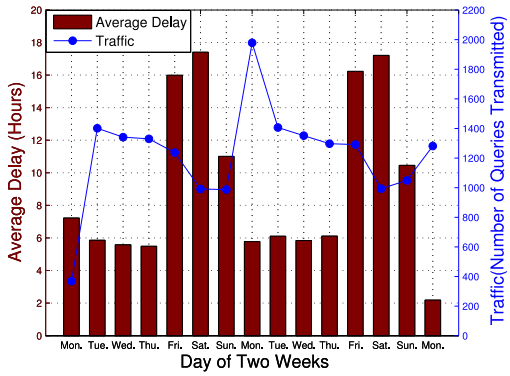
which indicates the overall ability of Node i to help a query in Category c to be answered. It serves as the routing metric to guide query delivery as to be discussed next.

2.2.2 Routing with Dynamic Redundancy Control

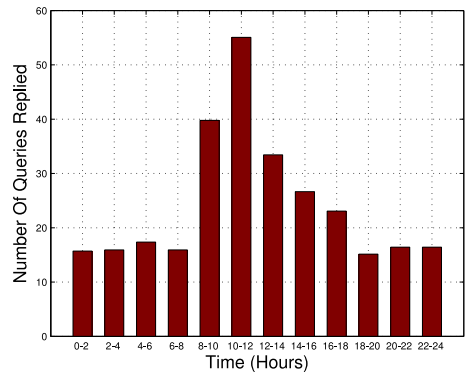
Based on the routing metric, i.e., reachable expertise, we now introduce the routing algorithm. The delivery of a query is guided by the aggregated reachable expertise, where the query is generally forwarded from the node with a lower aggregated reachable expertise to the node with a higher



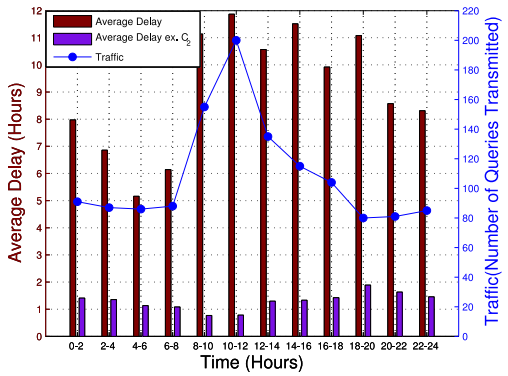
(a) Query replies (daily).



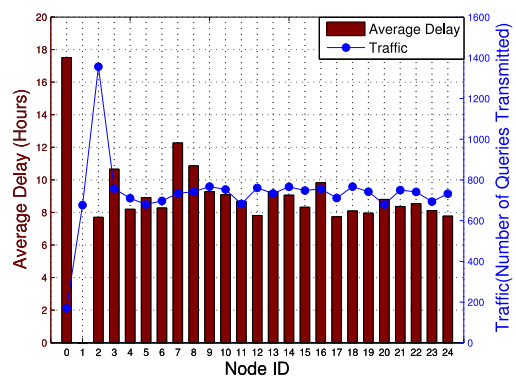
(b) Delay and traffic (daily).



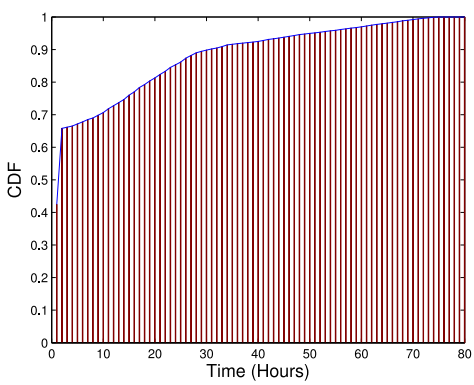
(c) Query replies (hourly).



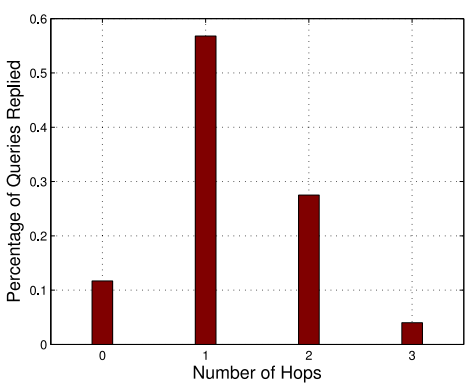
(d) Delay and traffic (hourly).



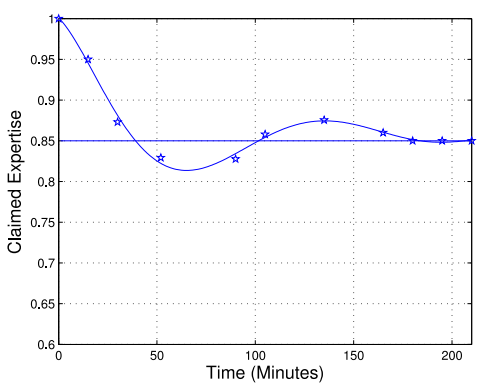
(e) Delay and traffic (nodes).



(f) Delay distribution.



(g) Path length distribution.



(h) Convergence of expertise.

Fig. 3. Experimental results.

one. In contrast to the conventional store-and-forward data transmission where a single copy of data is transmitted across the network, redundancy is often employed in opportunistic networks. While redundancy is not addressed in the analysis due to its intractability, it is important in practice to achieve the desired query delivery rate. Generally speaking, the higher the redundancy, the higher probability the query is answered successfully. However, redundancy must be properly controlled as excessive redundancy may exhaust network capacity and thus degrade the performance.

A naive approach is to create a fixed amount of redundancy for each query. For example, a predetermined number of copies of the query can be created and distributed to other nodes in the network. This approach, however, is often inefficient, because the effectiveness of redundancy depends on the nodes that receive, carry and forward the query. In an extreme case, all redundant copies of the query may be transmitted and carried by the nodes that have little chance to meet the node(s) that can answer the query and thus become ineffective. As a matter of fact, the effectiveness of redundancy highly depends on the reachable expertise of the nodes that carry the redundant copies. To this end, we introduce a parameter to dynamically reflect the "effective redundancy".

More specifically, the proposed routing algorithm with dynamic redundancy control is outlined below. Let R_i^q denote the redundancy of Query q as observed by Node i. The parameter is the estimated probability that at least one copy of the query is answered by any other nodes in the network. It is maintained and updated in a distributed way. Assume Query q is in Category c. R_i^q is initialized to zero when the query is created and subsequently updated during its transmission. Since communication opportunity is low, transmission is often between two nodes only. If more than two nodes are within communication range, we assume an underlying medium access control protocol that randomly selects one node as the sender. Therefore we consider a general scenario where Node i meets Node j in the following discussions.

First, Nodes i and j exchange their k-hop reachable expertise and update their aggregated reachable expertise according to Eqs. (6) and (7).

Then, Node i fetches the query with the lowest redundancy in its queue. The queue holds the queries that Node i creates or receives from other nodes. It is sorted according to the redundancy level such that the query with the lowest redundancy (denoted as Query q in Category c) is at the head of the queue. If Node j has a high expertise for queries in Category c (i.e., $E_j^c \geq \alpha$ where α is a predefined constant), it directly answers the query by creating and sending a query reply to the query issuer. Since the destination of the query reply (i.e., the query issuer) is known, it can be delivered via any existing routing protocol for opportunistic networks [11], [12], [13], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [40], [41], [42], [43]. We adopt the scheme proposed in [24] in our implementation. Note that the answer of Node j is not always satisfactory. It has a probability of E_j^c to be satisfied by the query issuer. Therefore, Node i removes the query from its queue with a probability of E_j^c.

Otherwise, if Node j cannot answer the query directly (i.e., $E_j^c < \alpha$), Node i checks the redundancy of Query q. If $R_i^q \geq \beta$ where β is the desired query delivery probability, it implies that high enough redundancy has been created for the query. Thus no action will be taken. Node i simply holds the query until it meets a node that can directly answer the query or the query must be dropped due to queue overflow. An overflow happens when a new query is added into the queue which is already full. In this case, the query with the highest redundancy (i.e., at the end of the queue) is dropped.

If $R_i^q < \beta$, the query should be further propagated. But it is transmitted to Node j only if $AE_i^c < AE_j^c$ (i.e., the latter has a better chance to deliver the query). This transmission creates two copies of the query, each sharing partial responsibility to get the answer. The redundancies for the two copies are assigned as follows:

$$R_j^q \leftarrow 1 - \big(1 - R_i^q\big)\big(1 - E_i^c(0)\big), \tag{8}$$

$$R_i^q \leftarrow 1 - \big(1 - R_i^q\big)\big(1 - E_j^c(0)\big). \tag{9}$$

In both formulas, $(1 - R_i^q)$ denotes the estimated probability that none of other nodes (except Nodes i and j) can get the answer for Query q, and $(1 - E_i^c(0))$ and $(1 - E_j^c(0))$ give the probability that Node i and Node j cannot directly answer the query. Therefore the updated R_i^q (or R_j^q) indicates the probability that at least one copy of the query can be answered by other nodes except Node i (or Node j).

The transmission of Query q continues upon future communication opportunities until, as discussed earlier, the query is answered by a node or dropped due to queue overflow. Upon receiving the query reply, the query issuer evaluates it and constructs a feedback packet, which is delivered to the node that answers the query, again, via an existing routing protocol for opportunistic networks. The latter then updates its expertise according to Eq. (5).

3 Prototype and Experiment

To demonstrate the feasibility and efficiency of the proposed data query protocol and to gain useful empirical insights, we have carried out a testbed experiment using off-the-shelf Dell Streak tablets. In this section, we first introduce our testbed setup and then present experimental results.

3.1 Prototype and Testbed Setup

We have developed a prototype system by using Dell Streak 5 and 7 tablets that are of the smartphone/tablet PC hybrid operating on Android 2.2. The communication between the tablets is enabled via Bluetooth. A Streak tablet has 16 GB internal storage adequate to keep large amounts of experimental data. We have implemented our proposed data query protocol by using standard Android APIs, closely following the description in the previous section. In order to save power, each node initiates neighbor discovery once every a random interval (between 5 to 10 minutes).

Our experiment involves twenty five volunteers including faculty members and students. They are marked as Node 0 to 24. In the experiment, we define three categories

TABLE 1
Results under Experiment

	Query Rate	Delay	Ave. Replies	Ave. Copies
0-hop	0.34	4.56h	10	1
1-hop	0.73	7.52h	30	2
2-hop	0.96	8.83h	42	2
3-hop	0.86	9.42h	56	4
No Feedback	0.83	9.12h	50	3
Flooding	0.39	4.68h	13	7
Gossip	0.52	7.28h	21	5
Willingness	0.51	6.23h	17	6
Spray and Wait	0.46	5.67h	12	6
Social-based	0.68	9.68h	58	5

of queries, i.e., history, science and arts (which are named Category 0, 1 and 2, respectively). Each participant has a claimed initial expertise for answering queries in each category and generates twelve queries per day in randomly chosen categories. Note that the initial expertise is not accurate. The true expertise is arbitrarily set by letting a small set of nodes to have an expertise of 1 to answer queries in each category. More specifically, Nodes 3, 13, and 19 can answer queries in Category 0; Nodes 4, 14, and 20 can answer queries in Category 2; and Node 1 can answer queries in any category. Other nodes initialize expertise to be $(0-1)$ randomly and learn and update their aggregated expertise during the experiment. The experiment had run for fifteen days, starting from Monday 16:00 p.m. in the first week to Monday 17:00 p.m in the third week.

We compare several variations of the proposed scheme and related schemes. In the following discussions, zero-hop, one-hop, two-hop, and three-hop stand for the proposed scheme that allows up to zero hop, one hop, two hops, and three hops relaying, respectively; No Feedback means the proposed scheme (with two-hop relay) but without the feedback mechanism to rectify expertise; Flooding is the simple flooding scheme for query delivery; Gossip [44] considers multiple categories and assigns the queries in each category a transmission probability for data transmission; Willingness [45] is a scheme that a query is delivered based on willingness, which is the degree to which a node actively engages in trying to re-transmit a query; Spray and Wait [33] is considered as a baseline opportunistic delivery protocol; Social-based [16] is a social-based routing scheme.

We are primarily interested in two parameters: (1) the success query rate, i.e., the ratio of successfully answered queries to the total generated queries, and (2) the query delay which is the period from the time when a node generates the query to the time when it receives the answer.

3.2 Experimental Results

Table 1 shows the overall performance of different schemes. The two-hop scheme achieves the highest query rate. It is not surprising to find the one-hop scheme with a lower query rate since a node merely tries to answer the queries via up to one hop relay. The zero-hop scheme has the lowest query rate as a query can be answered only when the query issuer meets the data provider directly. On the other hand, it seems anti-intuitive that allowing a longer relay path (e.g., the three-hop scheme) leads to a negative gain. But this is reasonable because excessive redundancy is created when too many nodes are involved in relaying queries, subsequently overloading the network and resulting in degraded performance. For a similar reason, the Flooding scheme has a even lower query rate given its extremely high redundancy. Under the No Feedback scheme, the inaccurate expertise is not rectified, resulting in misleading reachable expertise and thus lower query rate. Gossip [44] considers multiple categories and assigns the queries in each category a transmission probability for data transmission. However, as a gossiping approach, its data transmission is randomized. Therefore a query is often answered and carried by nodes with insufficient expertise, thus inducing many non-satisfactory replies. Willingness [45] is a scheme that a query is delivered based on willingness, which is the degree to which a node actively engages in trying to re-transmit a query. The willingness does not reflect the expertise based on which a node replies queries, therefore the nodes are not helpful for each other to carry queries to nodes with sufficient expertise. We also compare with Spray and Wait [33] which is considered as a baseline opportunistic delivery protocol. [33] fixes the number of copies for each query which limits the queries to go through correct paths to be replied by nodes with sufficient expertise, making query rate even lower. Social-based [16] exploits a distributed community partitioning algorithm to divide a DTN into smaller communities. For intra-community communication, a utility function convoluting social similarity and social centrality with a decay factor is used to choose relay nodes. For inter-community communication, the nodes moving frequently across communities are chosen as relays to carry data to destination efficiently. Although Zhu et al. [16] introduces a solution for DTNs which leverages social properties and mobility characteristics of users, it is not truly applicable for the data query in MASONs, because when a node issues a query, it is often unaware of the nodes that have sufficient expertise to answer the query. The cost is prohibitively high to construct a structure to index data and data providers like P2P networks. It is obviously inefficient either to frequently flood queries, which are expensive and often considered spams. It is not surprising that our proposed scheme has better performance than Social-based, since Social-based does not capture the inherent features for the query delivery in MASONs, hence the nodes are not helpful for each

other by making the correct decisions to carry queries to satisfactory nodes.

In general, when more hops are allowed in relaying queries, the overhead increases, because a query is more aggressively propagated. As a result, more copies of the query are transmitted in the network and the query issuer often receives more replies. At the same time, since a query may potentially go through a longer path to reach the data provider, the average delay also increases. Compared with the two-hop scheme, No Feedback has longer delay and more number of replies because incorrect expertise often leads the queries to wrong routes.

More than 96 percent queries are answered successfully. The unanswered queries are all generated during the final hours of the experiment. Fig. 3a illustrates the number of queries answered on each day of the experiment. As can be seen, the results vary among days, reflecting the moving patterns of the participants. More queries are answered during weekdays than weekends due to the lower interactive activities of students and faculty on Saturday and Sunday. In fact, many queries cannot be answered during weekends and have to wait until Monday of the next week. This explains the peak on the second Monday. It is worth mentioning that the first and the third Monday are not the whole days, hence the number of answered queries is less than the second Monday. The activity pattern is also evidenced by the delay variation shown in Fig. 3b. Queries generated in weekends have longer delay compared with those in weekdays. The delay of queries generated on Friday is also high because no classes are scheduled on Friday afternoon and many offices are closed after 1:00 p.m.. Fig. 3b also shows the total traffic in the network, which follows a similar pattern of nodal activities.

Figs. 3c and 3d further zoom in to show the results in each hour of a day. The data are averaged over 15 days. Both the network traffic and the number of answered queries are high during daytime and low at night, which again shows the query heavily depends on the activity of students and faculty who carry the tablets. Likewise, we expect lower delay during daytime. However, the results are just the opposite (as depicted in Fig. 3d). Such anti-intuitive observation is due to the queries from a few nodes, which experienced extremely long delay that dominates the overall performance. In fact, most queries generated during daytime indeed have short delay. But a set of nodes (including Nodes 3-12) rely on a single node (Node 2) to carry their queries in Category 2 to corresponding data providers. Such delivery happens around 9:00 a.m. daily. The queries generated after 9:00 a.m. must wait until the next day, thus inducing unusually long latency that significantly elevates the overall average. If we exclude such queries (see the lower purple bars in Fig. 3d), the average delay becomes much lower, and the daytime delay is generally shorter than that during night.

The average delay and traffic of different nodes are illustrated in Fig. 3e. In general, delay and traffic vary among different nodes due to the randomness in nodal mobility and query generation and transmission. Node 0 has extremely poor connectivity (either directly or indirectly) to the nodes with high expertise, resulting in very long delay compared with other nodes. Contrarily, since Node 1 is able to answer all the queries of three categories, it has the minimum delay. In addition, Node 2 has the heaviest traffic load because it frequently meets other nodes, while Node 0 carries the least traffic due to few interactions between it and other nodes.

The delay distribution is shown in Fig. 3f. More than 65 percent queries are answered within two hours. The queries with longer delays are either generated by Nodes 3-12 as discussed above, or generated during weekends and thus cannot be replied until the next Monday. Fig. 3g illustrates the distribution of path length. All queries are answered within three hops. Fig. 3h shows the convergence of the claimed expertise to the ground truth. A node is chosen as an example, while similar results are observed in other nodes as well. As we can see, the feedback mechanism effectively adjusts the node's expertise, gradually approaching to the true value within a few hours.

4 Simulation Results

Besides the experiment discussed above, extensive simulations are carried out to learn the performance trend of the proposed data query algorithm under various network settings, which are not practical to evaluate by using lab equipments. The simulation codes are extracted from our prototype implementation, and the simulation results are obtained under real-world traces and power-law mobility model. Each node maintains a maximum queue size of $1{,}000$.

4.1 Simulation under Haggle Trace

We have evaluated our proposed scheme under several real-world traces available at CRAWDAD [46]. Table 2 shows the results based on Haggle trace [39], which includes 98 participants carrying small devices (iMotes) during Infocom 2006. We run the simulation for a period of $342,916$ seconds (or about 4 days). Each node generates 1.08 queries per hour. The queries fall into five categories, and each category is associated with three expert nodes that can provide satisfactory replies. Similar to the results in Table 1, Table 2 shows the results under Haggle trace.

The distributions of query rate and delay are illustrated in Fig. 4. About 90 percent of nodes can achieve a query rate of 80 percent or higher under the proposed scheme. At the same time, more than half of the queries are answered within an hour.

4.2 Simulation under Power-Law Mobility Model

Besides the above results based on Haggle trace, we have carried out simulations under power-law mobility model, which enables convenient study of performance trend with the variation of several network parameters. More specifically, we simulate an area that is partitioned into a grid of 20×20 cells. Each node is associated with a randomly-chosen home cell, in which it initially resides. In a time slot, it may move in one of the four directions, i.e., up, down, left and right, or stay in its home cell. Let $P_i(x)$ denote the probability for Node i to be at Cell x. $P_i(x) = k_i(\frac{1}{d_i(x)})^{\sigma}$ where k_i is

TABLE 2
Results under Haggle Trace

	Query Rate	Delay	Ave. Replies	Ave. Copies
0-hop	0.31	1.58h	9	1
1-hop	0.68	2.62h	26	2
2-hop	0.88	3.06h	39	3
3-hop	0.79	3.25h	52	4
No Feedback	0.76	3.16h	46	3
Flooding	0.36	1.61h	12	6
Gossip	0.46	2.56h	20	5
Willingness	0.45	2.16h	16	5
Spray and Wait	0.39	1.98h	11	5
Social-based	0.62	3.32h	56	4

a constant, σ is the exponent of the power-law distribution and $d_i(x)$ denotes the distance between Cell x and Node i's home cell. The default network parameters include a network of 100 nodes, a σ of 2, 10 categories of queries, five experts per category, and a generation rate of 0.02 queries per time unit per node.

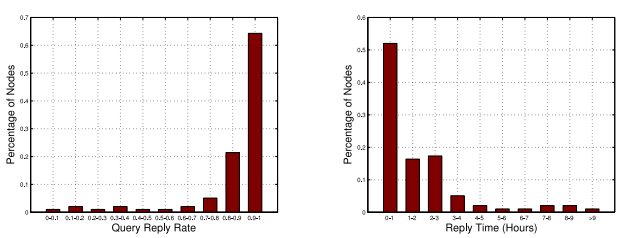
In an opportunistic network, the communication capacity highly depends on the meeting opportunities among mobile nodes. As shown in Fig. 5a, query reply rate grows with the increase of the network density, because the nodes have more opportunities to meet each other and exchange their queries. Fig. 5b depicts the impact of the power-law factor σ. When σ is large, the nodes tend to stay in their home cells, i.e., have low mobility, resulting in small probabilities to meet each other and consequently small network capacity. Therefore the query reply rate is low. When σ is extremely large, the query reply rate may approach as low as zero. On the other hand, when σ is small, the nodes have uniform mobility, i.e., similar probabilities to visit all cells and accordingly similar routing metric (i.e., k-hop reachable expertise), rendering routing ineffective. Under the simulated network setting, $\sigma = 2$ results in the best performance.

The impact of traffic load is illustrated in Fig. 5c. While the query reply rate keeps stable at the beginning under all schemes, it starts to drop when the generation rate exceeds 0.03. In general, with a higher query generation rate, the overall traffic load increases, resulting in more frequent queue overflow and consequently lower query reply rate.

As discussed in Section 2, a threshold β is employed for dynamic redundancy control. A larger β allows more redundancy to be created, aiming to achieve a higher query reply rate. However, if β is too large, the excessive redundancy degrades the utilization of communication and storage resources and lowers the overall performance accordingly (see the Fig. 5d). Fig. 5e shows that a higher query reply rate is achieved with the increase of queue size, because more queries and replies can be kept in the queue until they are delivered. The number of experts for each category is also studied in this work. As shown in Fig. 5f, more experts for a category result in higher query reply rate because more nodes can answer the queries in this category.

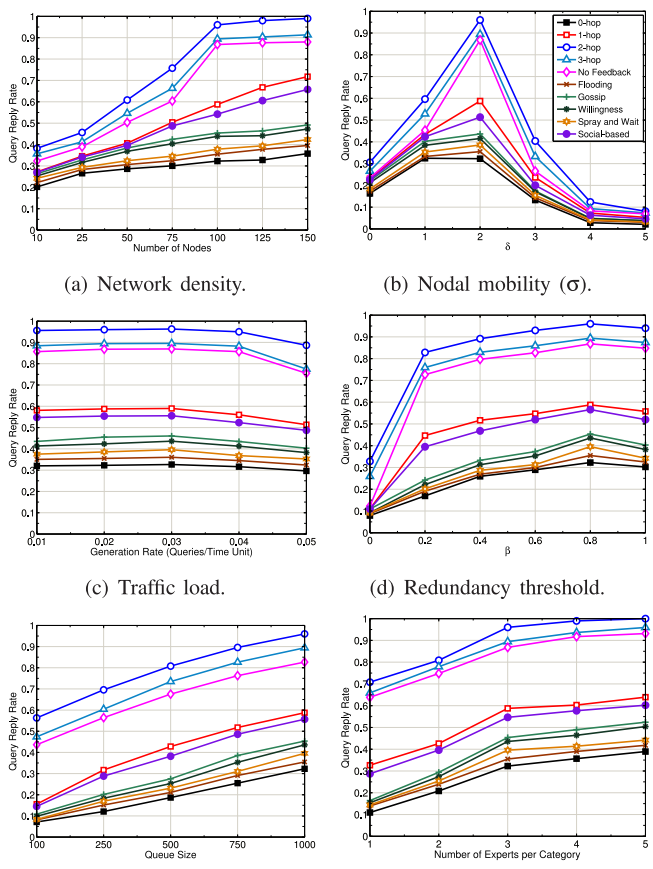
5 Conclusion

We have studied the problem of data query in a Mobile Adhoc SOcial Network, aiming to determine an optimal transmission strategy that supports the desired query rate within



(a) Query rate distribution. (b) Delay distribution.

Fig. 4. Distribution under Haggle trace.



(a) Network density. (b) Nodal mobility (σ).

(c) Traffic load. (d) Redundancy threshold.

(e) Queue size. (f) Number of experts/category.

Fig. 5. Performance trend under power-law mobility model.

a delay budget and at the same time minimizes the total communication cost. We have proposed a centralized optimization model that offers useful theoretic insights and developed a distributed data query protocol for practical applications. To demonstrate the feasibility and efficiency of the proposed scheme and to gain useful empirical insights, we have carried out a testbed experiment by using 25 off-the-shelf Dell Streak tablets for a period of 15 days. Moreover, we have run extensive simulations to learn the performance trend under various network settings, which are not practical to build and evaluate in laboratories.

REFERENCES

[1] B. Yang and A. Hurson, "A content-aware multimedia accessing model in ad hoc networks," in *Proc. 11th Int. Conf. Parallel Distrib. Syst.*, 2005, pp. 613–619.

[2] C. Avin and C. Brito, "Efficient and robust query processing in dynamic environments using random walk techniques," in *Proc. 3rd Int. Symp. Inf. Process. Sens. Netw.*, 2004, pp. 277–286.

[3] N. Chang and M. Liu, "Controlled flooding search in a large network," *IEEE/ACM Trans. Netw.*, vol. 15, no. 2, pp. 436–449, Apr. 2007.

[4] T. Hara, "Effective replica allocation in ad hoc networks for improving data accessibility," in *Proc. IEEE 20th Annu. Joint Conf. Comput. Commun.*, 2001, pp. 1568–1576.

[5] W. W. Terpstra, J. Kangasharju, C. Leng, and A. P. Buchmann, "BubbleStorm: Resilient, probabilistic, and exhaustive peer-to-peer search," in *Proc. ACM Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2007, pp. 49–60.

[6] A. N. Shiferaw, V.-M. Scuturici, and L. Brunie, "Interest-awareness for information sharing in MANETs," in *Proc. 11th Int. Conf. Mobile Data Manage.*, 2010, pp. 342–347.

[7] Z. Li, H. Shen, G. Liu, and J. Li, "SOS: A distributed mobile QA system based on social networks," in *Proc. IEEE 32nd Int. Conf. Distrib. Comput. Syst.*, 2012, pp. 627–636.

[8] P. Hui, J. Leguay, J. Crowcroft, J. Scott, T. Friedmani, and V. Conan, "Osmosis in pocket switched networks," in *Proc. 1st Int. Conf. Commun. Netw. China*, 2006, pp. 1–6.

[9] J. Fan, J. Chen, Y. Du, P. Wang, and Y. Sun, "DelQue: A socially aware delegation query scheme in delay-tolerant networks," *IEEE Trans. Vehicular Technol.*, vol. 60, no. 5, pp. 2181–2193, Jun. 2011.

[10] Z. Yang, T. Ning, and H. Wu, "Distributed data query in intermittently connected passive RFID networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 10, pp. 1972–1982, Oct. 2013.

[11] A. Mei, G. Morabito, P. Santi, and J. Stefa, "Social-aware stateless forwarding in pocket switched networks," in *Proc. IEEE Conf. Comput. Commun.*, 2011, pp. 251–255.

[12] W. Gao and G. Cao, "User-centric data dissemination in disruption tolerant networks," in *Proc. IEEE Conf. Comput. Commun.*, 2011, pp. 3119–3127.

[13] Z. Li and H. Shen, "SEDUM: Exploiting social networks in utility-based distributed routing for DTNs," *IEEE Trans. Comput.*, vol. 62, no. 1, pp. 83–97, Jan. 2011.

[14] E. M. Daly and M. Haahr. (2007). "Social network analysis for routing in disconnected delay-tolerant MANETs" *Proc. 8th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, pp. 32–40. [Online]. Available: http://doi.acm.org/10.1145/1288107.1288113

[15] P. Hui, J. Crowcroft, and E. Yoneki. (2009).,"Bubble Rap: Social-based forwarding in delay tolerant networks" in *Proc. 9th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, pp. 241–250 [Online]. Available: http://doi.acm.org/10.1145/1374618.1374652

[16] K. Zhu, W. Li, and X. Fu, "SMART: A social and mobile aware routing strategy for disruption tolerant networks," *IEEE Trans. Vehicular Tech.*, vol. PP, no. 99, pp. 1–1, 2014.

[17] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on the design of opportunistic forwarding algorithms," in *Proc. IEEE Conf. Comput. Commun.*, 2006, pp. 1–13.

[18] M. Kim, D. Kotz, and S. Kim, "Extracting a mobility model from real user traces," in *Proc. IEEE Conf. Comput. Commun.*, 2006, pp. 1–13.

[19] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Performance analysis of mobility-assisted routing," in *Proc. 7th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2006, pp. 49–60.

[20] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. ACM Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2003, pp. 27–34.

[21] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem," in *Proc. ACM Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2007, pp. 373–384.

[22] D. Gunawardena, T. Karagiannis, A. Proutiere, E. Santos-Neto, and M. Vojnovic, "Scoop: Decentralized and opportunistic multicasting of information streams," in *Proc. 11th ACM Int. Conf. Mobile Comput. Netw.*, 2011, pp. 169–180.

[23] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Efficient routing in intermittently connected mobile networks: The single-copy case," *IEEE/ACM Trans. Netw.*, vol. 16, no. 1, pp. 77–90, Feb. 2008.

[24] P. Juang, H. Oki, Y. Wang, M. Martonosi, L.-S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with Zebranet," in *Proc. Annu. Conf. Archit. Support Program Languages Operating Syst.*, 2002, pp. 96–107.

[25] T. Small and Z. J. Haas, "The shared wireless infostation model—A new ad hoc networking paradigm," in *Proc. 4th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2003, pp. 233–244.

[26] B. Burns, O. B. Brian, and N. Levine, "MV routing and capacity building in disruption tolerant networks," in *Proc. IEEE Conf. Comput. Commun.*, 2005, pp. 398–408.

[27] N. Banerjee, M. D. Corner, D. Towsley, and B. N. Levine, "Relays, base stations, and meshes: Enhancing mobile networks with infrastructure," in *Proc. 14th ACM Int. Conf. Mobile Comput. Netw*, 2008, pp. 81–91.

[28] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in *Proc. ACM SIGCOMM Workshop Delay-Tolerant Netw.*, 2005, pp. 244–251.

[29] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell, "The BikeNet mobile sensing system for cyclist experience mapping," in *Proc. 5th Int. Conf. Embedded Netw. Sens. Syst.*, 2007, pp. 87–101.

[30] P. Klasnja, B. L. Harrison, L. LeGrand, A. LaMarca, J. Froehlich, and S. E. Hudson, "Using wearable sensors and real time inference to understand human recall of routine activities," in *Proc. 10th Int. Conf. Ubiquitous Comput.*, 2008, pp. 154–163.

[31] E. Miluzzo, N. D. Lane, K. Fodor, R. A. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: The design, implementation and evaluation of the CenceMe application," in *Proc. 6th ACM Conf. Embedded Netw. Sens. Syst.*, 2008, pp. 337–350.

[32] P. Mohan, V. Padmanabhan, and R. Ramjee, "Nericell: Rich monitoring of road and traffic conditions using mobile smartphones," in *Proc. 6th ACM Conf. Embedded Netw. Sens. Syst.*, 2008, pp. 323–336.

[33] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *Proc. ACM SIGCOMM Workshop Delay-Tolerant Netw.*, 2005, pp. 252–259.

[34] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 7, no. 3, pp. 19–20, 2003.

[35] S. Jain, M. Demmer, R. Patra, and K. Fall, "Using redundancy to cope with failures in a delay tolerant network," in *Proc. ACM Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2005, pp. 109–120.

[36] A. A. Hanbali, P. Nain, and E. Altman, "Performance of ad hoc networks with two hop relay routing and limited packet lifetime," *Performance Eval.*, vol. 65, no. 6, pp. 463–483, 2006.

[37] X. Tie, A. Venkataramani, and A. Balasubramanian, "R3: Robust replication routing in wireless networks with diverse connectivity characteristics," in *Proc. 17th Annu. Int. Conf. Mobile Comput. Netw.*, 2011, pp. 181–192.

[38] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497–520, 1960.

[39] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. (2009). "CRAWDAD trace cambridge/haggle/imote/infocom2006 (v. 2009-05-29)," [Online]. Available: http://crawdad.cs.dartmouth.edu/

[40] Y. Wang and H. Wu, "DFT-MSN: The delay fault tolerant mobile sensor network for pervasive information gathering," in *Proc. IEEE Conf. Comput. Commun.*, 2006, pp. 1–12.
[41] Y. Wang and H. Wu, "Delay/fault-tolerant mobile sensor network (DFT-MSN): A new paradigm for pervasive information gathering," *IEEE Trans. Mobile Comput.*, vol. 6, no. 9, pp. 1021–1034, Sep. 2007.
[42] Y. Wang, H. Wu, F. Lin, and N.-F. Tzeng, "Cross-layer protocol design and optimization for delay/fault-tolerant mobile sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 5, pp. 809–819, Jun. 2008.
[43] H. Dang and H. Wu, "Clustering and cluster-based routing protocol for delay-tolerant mobile networks," *IEEE Trans. Wireless Commun.*, vol. 9, no. 6, pp. 1874–1881, Jun. 2010.
[44] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-based ad hoc routing," *IEEE/ACM Trans. Netw.*, vol. 14, no. 3, pp. 479–491, Jun. 2006.
[45] K. A. Harras and K. C. Almeroth, "Controlled flooding in disconnected sparse mobile networks," *Wireless Commun. Mobile Comput.*, vol. 9, no. 1, pp. 21–33, 2009.
[46] [Online]. Available: http://www.crawdad.org/, 2014.

Yang Liu (S'11) received the BE degree in electrical engineering and its automation, and the ME degree in control theory and control engineering from Harbin Engineering University, China, in 2008 and 2010, respectively. He has been working toward the PhD degree in computer engineering at the Center for Advanced Computer Studies, University of Louisiana at Lafayette, since 2011. 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.

Yanyan Han received the BS degree in electronic information engineering from Shandong University, Jinan, China, in 2008, and has been working toward the PhD degree in computer science at the Center for Advanced Computer Studies (CACS), University of Louisiana at Lafayette (UL Lafayette), since 2011. Her current research interests include mobile ad hoc networks and delay-tolerant networks.

Zhipeng Yang (S'07) received the BS and MS degrees in computer science from Tianjin University, China, in 2001 and 2004, respectively. He has been working toward the PhD degree in computer science at the Center for Advanced Computer Studies (CACS), University of Louisiana at Lafayette (UL 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.

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 interests include 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 the UL Lafayette Distinguished Professor Award in 2011. He is a member of the IEEE.