Social-Aware Data Diffusion in Delay Tolerant MANETs

Yang Zhang, Wei Gao, Guohong Cao, Tom La Porta, Bhaskar Krishnamachari, and Arun Iyengar

Abstract Most existing mobility-assisted data access techniques in delay tolerant mobile ad hoc networks (DT-MANETs) are designed to disseminate data to one or several particular destinations. Different from these works, we study the *data diffusion* problem which diffuses data among all moving nodes so that the nodes that are interested in this data item can get it easily either from their encountered friend nodes or stranger nodes. To reduce the data access delay, we introduce four social-aware data diffusion schemes based on the social relationship and data similarity of the contacts. We also provide solutions to quantify data/interest similarity and to determine whether two nodes are friends or strangers. Theoretical models are developed to analyze the data diffusion process and compare the performance of the four proposed diffusion schemes in terms of diffusion speed and query delay. We use real traces of human contacts to emulate data diffusion under different schemes. Both theoretical analysis and experimental results imply an interesting fact: to achieve better diffusion performance, each node should first diffuse the data similar to their common interests when it meets a friend, and first diffuse the data different to their common interests when it meets a stranger.

Yang Zhang, Wei Gao, Guohong Cao and Tom La Porta
Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA
e-mail: yangzhan,wxg139,gcao,tlp@cse.psu.edu

Bhaskar Krishnamachari
Department of Electrical Engineering, University of Southern California, Los Angeles, CA
e-mail: bkrishna@usc.edu

Arun Iyengar
IBM T.J. Watson Research Center, Hawthorne, NY
e-mail: aruni@us.ibm.com

1 Introduction

With the rapid adoption of mobile hand-held devices (e.g., PDAs, Bluetooth enabled mobile phones, active RFID tags, etc.), more and more people use them to diffuse, query and share interesting data among themselves without any network infrastructure support. Such a community-wide (or even city-wide) network formed by the mobile hand-held devices is an example of a *delay tolerant mobile ad hoc network (DT-MANET)* [1].

Due to the low node density and unpredictable network topology, routing paths in DT-MANETs may be frequently disconnected. To deal with such problems, *mobility-assisted data access* techniques have been exploited, where a node physically carries data for some time until it moves within the communication range of some other node (i.e., *contact*) [2, 3, 4]. Then, it decides whether to propagate the data to the new contact or not based on some algorithm. Existing algorithms such as *SimBet Routing* [5], *BUBBLE Rap* [6], *SocialCast* [7], *SOLAR* [8], and *MaxProp* [9] are designed to forward data to one given destination. Since the source and destination may be faraway from each other, the delay for the destination to get the data from the source may be long.

One way to reduce the query delay is through *data diffusion*, where data is diffused throughout the network and replicated in advance. Subsequent queries can be served by any node with the data instead of being sent to the source node, and thus reducing the query delay. However, data diffusion is not free. In DT-MANETs, nodes diffuse data when they are in contact. Because the contact time is pretty short and the buffer size of each node is limited, the diffused data has to compete for the limited bandwidth and buffer space. Therefore, the diffusion decisions made by each node such as which data should be propagated first and which data should be replaced out of the buffer, affect the diffusion speed and the data access delay.

In this chapter, we study the performance of different data diffusion schemes in DT-MANETs. Our classification of these data diffusion schemes is based on social networks. Social networks exhibit the “homophily” phenomenon [10] which comes from the observation that individuals often befriend others who have similar interests, and hence perform similar actions and have a higher possibility to meet with each other. For example, two individuals who own the same kind of video game console are more likely to become friends and meet at game shops due to the common interest in games. Students from the same department are more likely to take the same courses and appear in the same lab or building. Therefore, the contact frequencies are probabilistically different between two friends and two strangers, and this difference should be taken into consideration when designing data diffusion schemes.

To study the effects of social networking on data diffusion, we are interested in answering the following questions: *How does the data diffusion scheme used affect the diffusion speed and the data access delay? How to design better data diffusion schemes based on social networking results?* To answer these questions, we introduce four possible social-aware data diffusion schemes and develop theoretical models to analyze their performance in terms of diffusion speed and query delay.

Based on the analysis, we discover an interesting result: to achieve better performance, when a node meets a new contact, if the new contact is a friend, it should first diffuse the data similar to their common interests; if the new contact is a stranger, it should first diffuse the data different from their common interest. We also provide solutions to determine if a new contact is a friend or stranger based on their common interests. To verify the theoretical results, we use two real social contact traces [11, 12] to emulate the data diffusion process under different schemes, and find that the experimental results are consistent with our theoretical analysis.

The rest of this chapter is organized as follows. Section 2 discusses the related works. Section 3 describes the application scenario, as well as four social-aware data diffusion schemes and their implementation techniques. In Section 4, we presents the theoretical analysis on diffusion speed and access delay of the four schemes. Performance evaluations are shown in Section 5. Finally, we conclude the chapter in Section 6.

2 Related Work

There have been several theoretical and empirical works on how social behavior can be used to improve the performance of data access in delay tolerant networks. PeopleNet [13] is a wireless virtual social network which mimics the way people seek information via social networking. It is simple and scalable for efficient information search in a distributed manner. However, it uses infrastructure to propagate data and queries, which is different from the peer-to-peer MANET scenario. *SimBet Routing* [5] studies the "*small-world*" phenomenon of human society and uses ego-centric centrality and its social similarity to guide data forwarding. Messages are forwarded towards the node with higher centrality. Similarly, *BUBBLE Rap* [6] focuses on community and social centrality, and nodes are structured into communities. High popularity nodes and community members of the destination are selected as relays. Ghosh *et al.* [8] have identified the orbital movement pattern of human being and relay nodes are chosen based on the places that they frequently visit. Similarly, Costa *et al.* [7] provide a routing framework using social interaction information in publish-subscribe systems and Gao *et al.* [14] study the social-aware multicasting issues in delay tolerant networks. Bai and Helmy [15] study the last encounter based routing protocol that utilizes encounter history to create time gradients for information diffusion in wireless networks. Furthermore, Gao and Cao [16] exploits transient contact patterns to improve the performance of data forwarding, and Li *et al.* [17] considers the selfishness property of social nodes in data forwarding. Although [5, 6, 8, 7, 14, 15, 16, 17, 18] have applied sociological knowledge to data dissemination in DT-MANETs, these works consider the problem of disseminating data to one pre-determined destination node. Unlike these existing works, data diffusion is not for settings with a specific destination.

The aforementioned works aim to find the most suitable relay node to increase the possibility of reaching the final destination. Miklas *et al.* [19], Karagiannis *et al.*

[20], Chaintreau *et al.* [4] and Wang *et al.* [21] study the social factor of delay tolerant networks in a different way. They analyze the distribution of inter-contact time between mobile devices and conclude that the inter-contact time follows the power law distribution or the exponential decay distribution. Further, Hsu *et al.* [22] analyze wireless users' behavioral patterns by extensively mining wireless network logs and discover that the size of distinct WLAN user group follows a power-law distribution. Besides, in the area of vehicular DT-MANETs, algorithms [23] and [24] have been proposed for finding the right relays for data forwarding and Kapadia *et al.* [25] consider the problem of optimizing the replication profile of content to minimize the aggregate average data access delay given knowledge of content popularity. In [26], the authors also present a cache replacement policy for finite buffers in a vehicular DT-MANET that takes into account the differing interests of vehicles in different geographic locations, but this work does not explicitly consider social interactions between users. Our work differs from these works in that we study how to use the social network results to improve data dissemination through social aware data diffusion schemes.

To the best of our knowledge, the closest studies to our work are ContentPlace [27] and PodNet [28]: when two nodes meet, they decide which data object to exchange based on the information gathered about nodes' interests. However, both [27] and [28] are designed for publish/subscribe services so they assume each node only fetches its interesting data (for single node in [28] or for single community in [27]). Our work, instead, studies the problem in a more general perspective where nodes can not only carry the data they are interested in, they can also buffer data they are not interested in, and forward them to other interested nodes in the future.

Data dissemination can be modeled as spreading of infectious disease. Disease spreading in fixed networks has been studied in the past [3]. [29] also analyzed the epidemic spreading in mobile networks. Different from the analysis in [3] and [29] where all nodes have the same moving and interest features, in our data diffusion scenario, both the interested nodes and uninterested nodes can help diffuse data and they have different meeting frequency and different interest preference. Therefore, different infection and immunization rates between interested and uninterested nodes are studied and the overall diffusion rates by both interested and uninterested nodes are investigated in our work, which adds complexity to the analysis

3 Social-Aware Data Diffusion

In this section, we first present the target application scenario that our work relies on. After that, we introduce social-aware data diffusion as four possible schemes based on a two-dimensional classification that combines both interest similarity and data similarity, and then provide the necessary implementation techniques for the similarity classification, which is based on the predefined threshold.

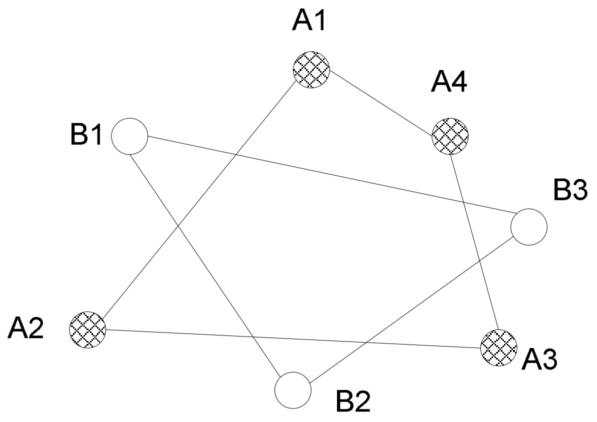


Fig. 1 Application scenario

3.1 Application Scenario

The application scenario we target is similar to the one used in ContentPlace [27] and PodNet [28] where we consider a number of mobile users whose devices cannot be encompassed by the conventional MANETs. Instead, communication is achieved by opportunistic pairwise contacts between users to exchange data objects. According to the homophily phenomenon, users with similar data interests have strong social relationships with each other. Further, people movements are governed by their social relationships, and by the fact that people with similar interests are also mostly bound to particular places (landmarks) that are associated with the interested topics [30, 31]. Therefore, users will spend most of their time and meet more friends at the landmarks they are bound to, and will also visit some other places occasionally and meet some stranger nodes at other places. This application scenario fits many existing service environments. For example, sports fans will spend more time in the sports related stores instead of cosmetics related stores when they are visiting an outlet. As a result, they are easy to meet other people with the same interests at those sports stores. In other words, two people in contact at these stores have a high possibility to have the similar interests.

Figure 1 gives a possible scenario with seven landmarks. Landmark A1-A4 are associated with the similar interest topics (e.g., sports related) and B1-B3 focus on another group of interest topics (e.g., cosmetics related). Then, people interested in sports are more likely to visit landmarks A1-A4. It is possible that they meet some people with different interests when they visit other landmarks sometimes. But in general, the proposition still holds that their contact rates with the people of similar interests are higher than that with other stranger people, and most of their encounters at the interested landmarks have similar interests as they have.

3.2 Diffusion Schemes

When two nodes meet each other, they exchange two lists. One is called the *interests list* which is the list of the interesting data; the other is the *data list* which records the data they are buffering. Based on these two lists, each node decides whether the encountered node can serve its query request and make further data diffusion decisions.

From the data perspective, all nodes can be divided into two non-overlapped groups, depending on whether if they are interested in one particular data or not. If one node is interested in the data, then this node is called the interested node of the data; otherwise, its is the uninterested node. Meanwhile, from the social network perspective, each node has two kinds of contacts: *friends* and *strangers*, where two nodes are friends if they have more similar interests, and they are strangers otherwise. According to homophily, friends usually share more common interests while strangers have less common interest (more details in Section 3.3). Therefore, we always hope that data can be diffused to interested nodes quickly so that most nodes can access their interesting data easily. However, the contact time can be very short in DT-MANETs and thus some data cannot be diffused between the two contacts. Also, the memory constraint limits the number of data items that a node can hold. Thus, we should carefully choose the most suitable data to diffuse and buffer first.

Without considering sociological knowledge, nodes diffuse data based on their own interests. Each node fetches and buffers interesting data from its contacts. Due to the bandwidth and buffer limitations, this solution has slow diffusion speed since each node only helps diffuse its own interesting data while neglects others. Another approach is to diffuse data randomly, where all data have the same opportunity to be diffused. However, this solution may diffuse much uninteresting data to some nodes, thus wasting the limited bandwidth and buffer space, and increasing the query delay.

With sociological knowledge, contacts can be categorized as friends or strangers and data can be categorized as being interesting or uninteresting. Thus, we have four possible data diffusion schemes by combining nodes' relationship and their interests in the data (as shown in Figure 2):

1. FsSd: When a node meets a new contact, if the new contact is a friend, it first sends the data of their common interest. These data items will be sorted based on their common interest (more details in Section 3.3). Each node sends the most similar data to its friend first, and then the second most similar data until the contact time is over. If the new contact is a stranger, it first sends the data different from their common interest. It will send the most different data first, and then the second most different data until the contact time is over.
   To summarize, it diffuses the most *S*imilar data between *F*riends, and diffuses the most *D*ifferent data between *S*trangers.
2. FsSs: it diffuses the most *S*imilar data between both *F*riends and *S*trangers.
3. FdSd: it diffuses the most *D*ifferent data between both *F*riends and *S*trangers.
4. FdSs: it diffuses the most *D*ifferent data between *F*riends and diffuses the most *S*imillar data between *S*trangers.

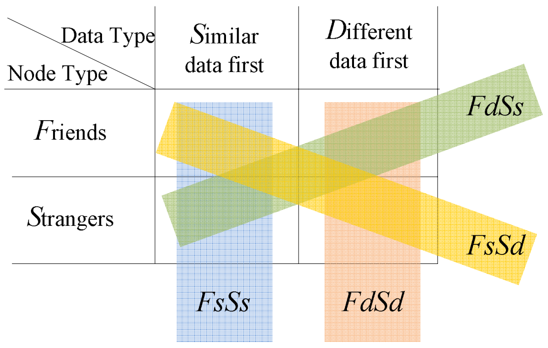


Fig. 2 Data diffusion schemes

If friends first diffuse the data that is most close to their common interests (Fs), their interesting data will have priority to be propagated and buffered between themselves. However, if friends diffuse the most different data first (Fd), the diffusion probability of their common interesting data will be low. On the other hand, if strangers first diffuse the data most different from their common interests as they meet (Sd), for one specific data, (notice that strangers share less interest similarity), the data still has a high probability to be diffused from its interested node to its uninterested node, and vice versa. Therefore, with FsSd, a node should be able to quickly diffuse data among its interested nodes, as well as between the interested nodes and its directly encountered uninterested nodes. Based on "homophily", friends have higher meeting frequency than strangers. If one data item can be buffered at more interested nodes, the query delay for this data item can be reduced. In this sense, FsSd may have the best diffusion performance. Before verifying this result through both theoretical analysis and experiments, we provide techniques for quantifying the interest/data similarity.

3.3 Measuring Similarity

To measure similarity, the first step is to formalize the description of data and query. Both data and query can be presented and indexed with resource representation techniques such as RDF (i.e., Resource Description Framework [32]) or WSDL (i.e., Web Services Description Language [33]) based on specific keyword attributes. In this chapter, to support complex data description, we associate each data with a sequence of keywords and define a mapping that preserves keyword similarity. The keywords are common words to describe data attributes such as "entertainment", "sport", "news", "travel", and etc. For example, music data may be labeled with "entertainment" and restaurant information can be indexed with "travel". Meanwhile, one data can have multiple attributes, thereby the same sport video might be labeled with both "entertainment" and "sport". Following this mapping method, all

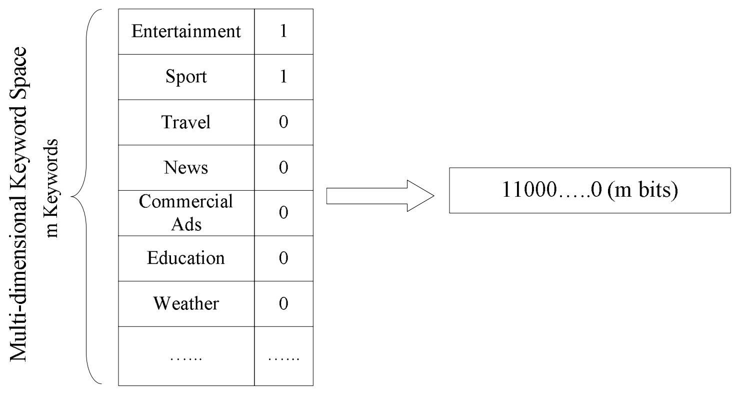


Fig. 3 Data description with m-dimensional attribute vector

attributes form a multi-dimensional keyword space so that the data is indexed by a multi-dimensional binary vector. If the data has one attribute, its corresponding bit in the vector is marked “1”; otherwise, it is marked “0”. For the simplicity of analysis and without loss of generality, we assume the attribute space is m-dimensional. Then each data can be described and indexed by a m-bit vector. Figure 3 demonstrates how to determine the vector of one data item, and Figure 4 shows an example of a keyword space. Similarly, query messages can be described in a similar way.

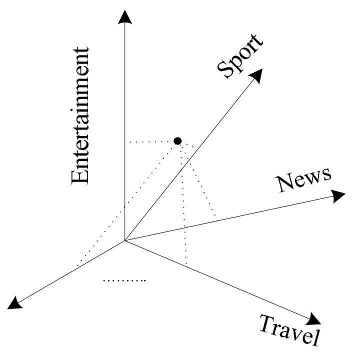


Fig. 4 A m-dimensional keyword space

Measuring interest similarity and data similarity for the classification of data diffusion schemes

The classification of our data diffusion schemes is based on the two-dimensional comparison of nodes’ interest similarity and data similarity.

First, in social-aware data diffusion, nodes make diffusion decisions based on their relationship (i.e., friends or strangers). Two friends share more common interests while two strangers have less interest similarity. Therefore, we need to estimate the interest similarity of two nodes to decide their relationship. Since node interest follows a probability distribution on different attributes, the interest similarity between two nodes should be calculated with two distributions. The Kullback-Leibler (K-L) divergence method [34] is used here to measure the difference between two probability distributions. If we use P_1 and P_2 to denote the discrete interest distributions of two nodes, the K-L divergence of P_2 from P_1 is defined to be

$$D_{KL}(P_1 \parallel P_2) = \sum_{i=1}^{m} P_1(i) log \frac{P_1(i)}{P_2(i)}$$

Therefore, the interest similarity of two nodes can be estimated as

$$\begin{aligned} SV_{dd} &= \frac{1}{D_{KL}(P_1 \parallel P_2)} \\ &= \frac{1}{\sum_{i=1}^{m} P_1(i) log \frac{P_1(i)}{P_2(i)}} \end{aligned} \tag{1}$$

Suppose FS_{thres} is the interest threshold to estimate the interest similarity of two nodes. If one node pair has a SV_{dd} smaller than FS_{thres}, they share few common interests so that they are strangers; otherwise, they are friends.

Note that the K-L divergence is not symmetric, which means $D_{KL}(P_1 \parallel P_2)$ is not necessarily equal to $D_{KL}(P_2 \parallel P_1)$. In this chapter we always use the interest distribution of the node with smaller ID as the first parameter (i.e., P_1) of the K-L divergence calculation and the node with larger ID as the second parameter (i.e., P_2).

Second, during each contact, nodes need to sort the data according to the data similarity to their common interests. Since the node's interests are presented by distributions and data objects are described by vectors. We need to compare the similarity between one vector and one discrete distribution. In this case, the similarity of one vector and one distribution can be calculated by their inner-product. Formally,

$$\begin{aligned} SV_{vd} &= \parallel \mathbf{V} \cdot \mathbf{P} \parallel \\ &= \sum_{i=1}^{m} v^i \times p_i \end{aligned} \tag{2}$$

where $\mathbf{V} = \langle v_1^1, v_1^2, ...v_1^m \rangle$ is the description vector of data V, and $\mathbf{P} = \langle p_1, p_2, ...p_m \rangle$ is the distribution vector of the discrete interest distribution P. With the calculation of SV_{vd} and an interest threshold IN_{thres}, each node can distinguish the data that is most similar or different to nodes' common interests and choose the most proper ones to diffuse.

4 Theoretical Analysis of Data Diffusion

In this section, we develop theoretical models to analyze the performance of data diffusion. In Section 4.1, we first study the case in which nodes have infinite buffer. Without buffer limitation, nodes spread data to as many contacts as possible and never remove data from their buffers. However, due to the limitation of the contact time, not all data can be diffused during each contact. Different decisions on diffusing similar or different data between friends and strangers still affect the performance of data diffusion. In Section 4.2, we consider the finite buffer case where some data items have to be replaced when the buffer is full.

4.1 The Infinite Buffer Case

The diffusion of each data item can be modeled as spreading of infectious disease. Disease spreading in fixed networks has been studied in the past [3]. [29] also analyzed the epidemic spreading in mobile networks. In the infectious disease model, one node is "infected" if it has the data buffered in its memory. The node is "susceptible" to infection if it does not have the data, but could potentially get the data from other nodes. Different from the traditional "susceptible-infected-recovered (S-I-R)" model [3, 29], in the infinite buffer data diffusion scenario, data is never deleted as long as it is buffered at some node. Therefore, all nodes follow a two-state compartmental S-I model. Meanwhile, both the interested nodes and the uninterested nodes can help diffuse the data. Therefore, the different infection rates between interested and uninterested nodes should be considered.

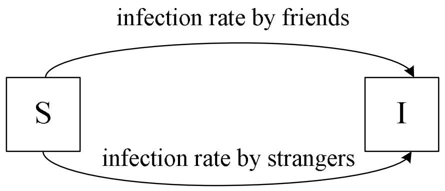


Fig. 5 Markov chain model of the S-I infectious disease with susceptible state and infected state (with infinite buffer)

First, for the interested nodes, as shown in Figure 5,

$$\begin{aligned} &total\ infection\ rate\ of\ interested\ node \\ &= infection\ rate\ by\ friends + infection\ rate\ by\ strangers \end{aligned}$$

We use susceptible state $S(t)$ and infected state $I(t)$ represents the number of nodes which are "susceptible" and "infected" in the system at time t, respectively.

Then, $I(t) = I_i(t) + I_u(t)$ and $S(t) = S_i(t) + S_u(t)$ where $I_i(t)$ and $S_i(t)$ are the numbers of "infected" and "susceptible" interested nodes, and $I_u(t)$ and $S_u(t)$ are the numbers of "infected" and "susceptible" uninterested nodes. β is the contact rate of one node to meet any other node[1], which consists of the contact rate among friends (β_f) and the contact rate among strangers (β_s). Further more, γ_f and γ_s are used to denote the data diffusion probability between two interested friends and from one interested node to any other uninterested stranger. Suppose there are N_i interested nodes in the system, then an interested node contacts $\beta_f(N_i - 1)$ other friends per unit time, of which $\frac{S_i}{N_i-1}$ do not yet have the data. The probability that the data will be exchanged to the encountered friend is γ_f. Therefore, the infection rate by friends can be estimated as

$$\begin{aligned}
&infection\ rate\ by\ friends\\
&= (\sharp\ of\ infected\ nodes)(contact\ rate\ of\ friends)\\
&\quad \times (infect\ probability\ of\ friends)(\sharp\ of\ susceptible\ nodes)\\
&= I_i(\beta_f \times (N_i - 1)) \times \gamma_f \times \frac{S_i}{N_i - 1}\\
&= I_i \beta_f S_i \gamma_f
\end{aligned}$$

Similarly, we can get the *infection rate by strangers* as $I_i \beta_s S_i \gamma_s$. Then, for a particular data item, the transition rate of any interested node from state S to state I becomes

$$\begin{aligned}
&total\ infection\ rate\ of\ interested\ node\\
&= infection\ rate\ by\ friends + infection\ rate\ by\ strangers\\
&= I_i \beta_f S_i \gamma_f + I_i \beta_s S_i \gamma_s\\
&= I_i S_i (\beta_f \gamma_f + \beta_s \gamma_s)
\end{aligned}$$

We are interested in the transient solution to the Markov chain in Figure 5. We can get $I_i(t)$ by solving the following first-order differential equation,

$$\begin{aligned}
\frac{dS_i}{dt} &= -I_i S_i (\beta_f \gamma_f + \beta_s \gamma_s)\\
\frac{dI_i}{dt} &= I_i S_i (\beta_f \gamma_f + \beta_s \gamma_s)\\
&= I_i (N_i - I_i)(\beta_f \gamma_f + \beta_s \gamma_s)\\
&= (\beta_f \gamma_f + \beta_s \gamma_s) N_i I_i - (\beta_f \gamma_f + \beta_s \gamma_s) I_i^2
\end{aligned}$$

This differential equation is separable and can be solved with the initial conditional $I_i(0) = 1$ to get the solution

[1] The contact rate does not mean the pairwise contact times for two specific nodes. Instead, it is the average number of contact for one node to meet any other node in the system.

$$I_i(t) = \frac{N_i}{1 + e^{-(\beta_f\gamma_f+\beta_s\gamma_s)N_i t}(N_i - 1)} \tag{3}$$

Similarly, based on the same S-I model, we can get the total infection rate of uninterested nodes as

$$\begin{aligned}
& total\ infection\ rate\ of\ uninterested\ node \\
&= infection\ rate\ by\ friends + infection\ rate\ by\ strangers \\
&= I_u\beta_f S_u\gamma_f' + I_u\beta_s S_u\gamma_s' \\
&= I_u S_u(\beta_f\gamma_f' + \beta_s\gamma_s')
\end{aligned}$$

where γ_f' and γ_s' are the diffusion probabilities between two uninterested friends and uninterested strangers.

If we use N_u to represent the number of uninterested nodes in the system, then the first infected uninterested node is expected to appear at time $\frac{1}{\beta_s\cdot N_u\cdot\gamma_s}$. Therefore, $I_u(t)$ can be approximated in the same way as $I_i(t)$ with a time offset of $\frac{1}{\beta_s\cdot N_u\cdot\gamma_s}$, i.e.,

$$I_u(t)=\begin{cases} 0 & t \le \frac{1}{\beta_s\cdot N_u\cdot\gamma_s} \\ \frac{N_u}{1+e^{-(\beta_f\gamma_f'+\beta_s\gamma_s')N_u(t-\frac{1}{\beta_s\cdot N_u\cdot\gamma_s})}(N_u-1)} & \text{else} \end{cases} \tag{4}$$

We use P_i and P_u to denote the probabilities of interested nodes and uninterested nodes to initiate the query. Then for one specific data, its expected query delay at time t, $E_Q(t)$, can be estimated as

$$\begin{aligned}
E_Q(t) &= E(query\ delay\ of\ interested\ node)\cdot P_i \\
&\quad + E(query\ delay\ of\ uninterested\ node)\cdot P_u
\end{aligned} \tag{5}$$

In particular, if the query is initiated by one interested node, this node can get the data either from its friends (according to “homophily”, they are also the interested nodes) or from its strangers (they are uninterested nodes).

First, if the data is from a friend node, because there are N_i interested nodes in the system and $I_i(t)$ interested nodes have the data at time t, the probability that the query node meets one friend and the friend has the data is $\frac{I_i(t)}{N_i-1}$. Meanwhile, as the query node can contact $\beta_f(N_i-1)$ interested friends per time unit, the average query delay of this case can be estimated as $\frac{N_i-1}{I_i(t)}\cdot\frac{1}{\beta_f(N_i-1)} = \frac{1}{I_i(t)\cdot\beta_f}$. Second, if the data is from a stranger node, the query node contacts $\beta_s(N_u)$ stranger per time unit and the probability that the contact nodes has the data is $\frac{I_u(t)}{N_u}$. Then we can get its average query delay as $\frac{N_u}{I_u(t)}\cdot\frac{1}{\beta_s(N_u)} = \frac{1}{I_u(t)\cdot\beta_s}$.

Therefore the expectation of the query delay of the interested node is the minimum query delay from either interested nodes or uninterested nodes, i.e.,

$$\begin{aligned}
& E(\ query\ delay\ of\ interested\ node) \\
&= \quad min\{\frac{1}{I_i(t)\cdot\beta_f}, \frac{1}{I_u(t)\cdot\beta_s}\}
\end{aligned} \tag{6}$$

However, if the query is initiated by one uninterested node, there are three kinds of nodes to serve the query: the friends of the node (who are also uninterested nodes), the uninterested strangers, and the interested strangers. Similarly, the expectation of the query delay of the uninterested node can be estimated as

$$\begin{aligned}
&E(query\ delay\ of\ uninterested\ node)\\
&= min\{\frac{N_i}{I_i(t)} \cdot \frac{1}{\beta_s N_i}, \frac{N_u-1}{I_u(t)} \cdot \frac{N_i}{N_u-1} \cdot \frac{1}{\beta_f N_i},\\
&\frac{N_u-1}{I_u(t)} \cdot \frac{N_u-N_i}{N_u-1} \cdot \frac{1}{\beta_s(N_u-N_i)}\}\\
&= min\{\frac{1}{I_i(t)\beta_s}, \frac{1}{I_u(t)\beta_f}, \frac{1}{I_u(t)\beta_s}\} \qquad (7)
\end{aligned}$$

Therefore, we can get

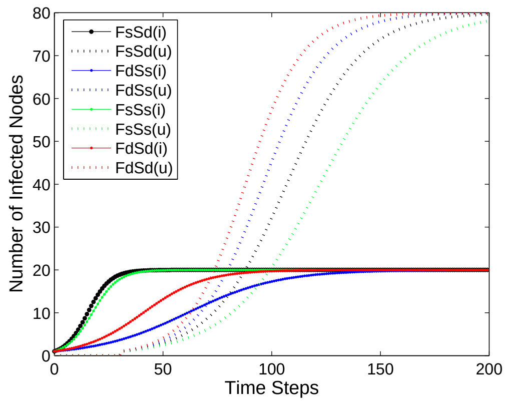
$$\begin{aligned}
E_Q(t) &= min\{\frac{1}{I_i(t)\beta_f}, \frac{1}{I_u(t)\beta_s}\} \cdot P_i\\
&+ min\{\frac{1}{I_i(t)\beta_s}, \frac{1}{I_u(t)\beta_f}, \frac{1}{I_u(t)\beta_s}\} \cdot P_u \qquad (8)
\end{aligned}$$

Different data diffusion schemes have different combinations of (γ_f, γ_s) and (γ'_f, γ'_s). For example in FsSd, suppose a node carries its interesting data. When this node meets its friend, most likely it will diffuse the data to its friend. When it meets a stranger the probability to diffuse this data between them is still high. This is because stranger nodes first choose the data that is most different to their common interests to diffuse. Also, two strangers have less interest similarity. If one node is interested in the data, its stranger might not be interested in it. Therefore both γ_f and γ_s are set to large values in FsSd.

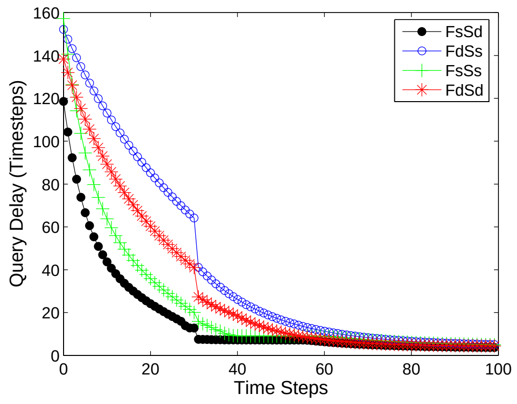
Suppose a node is carrying an uninteresting data item. When it meets a friend, its friend may also have no interest in this data, decreasing the diffusion possibility. If this node meets a stranger that is also not interested in the data, instead, it might diffuse this uninteresting data because the data is still different from the common interests of these two uninterested strangers and should have high diffusion priority according to FsSd. Consequently, γ'_f becomes small and γ'_s is still large. Similarly, we can set the values of (γ_f, γ_s) and (γ'_f, γ'_s) for the other three schemes as shown in Table 1.

Table 1 The setting of (γ_f, γ_s) and (γ'_f, γ'_s)

	γ_f	γ_s	γ'_f	γ'_s
FsSd	large	large	small	large
FdSs	small	small	large	small
FsSs	large	small	small	small
FdSd	small	large	large	large



(a) Infected Nodes



(b) Query Delay

Fig. 6 Numerical results based on the S-I analysis model ($\beta_f = 0.01$, $\beta_s = 0.002$, $N_i = 20$, $N_u = 80$)

Figure 6 shows some numerical results according to the analysis. Figure 6(a) depicts the number of infected nodes as a function of time. We use FsSd(i) and FsSd(u) to denote the number of infected interested nodes and infected uninterested nodes, respectively, under the FsSd scheme. The infected nodes of other three schemes are denoted similarly. As there is infinite buffer, all nodes should be infected after some amount of time. From the figure, we can see that at time 200, almost all nodes (20 interested nodes and 80 uninterested nodes) are infected. However, different diffusion schemes have different data diffusing speed. For example, at time 25, all the 20 interested nodes in FsSd(i) are infected, but it takes 130 time units for the 20 interested nodes to be infected in FdSs(i). Note that the diffusion speeds of FsSd and FsSs are slower than FdSs and FdSd among uninterested nodes. This is because both

FsSd and FsSs give high diffusion priority to the interested friends while sacrificing the diffusion opportunity of their uninteresting data.

Figure 6(b) investigates the query delay as a function of time for different diffusion schemes. We can see that FsSd has the shortest query delay. This is because FsSd diffuses the interesting data among its friends quickly. Homophily suggests that friends share more common interests and have a high meeting probability. Therefore quickly diffusing interesting data among friends results in lower query delay. FdSs has the lowest diffusion priority between interested friends and strangers, and thus it has the slowest diffusion speed and longest query delay. Notice that there is a sudden drop at about time 26. The sudden drop is due to the piecewise function (Eqn. (5)) that is used to estimate the appearance time of the first infected uninterested node. After an uninterested node gets the data, many queries might be served, and thus reducing the delay.

4.2 The Finite Buffer Case

With finite buffer, the analysis becomes more complicated since data may be removed from the buffer. In this case, the S-I model should be replaced by the S-I-S model in which infected nodes return to the susceptible state on recovery because they are not against reinfection.

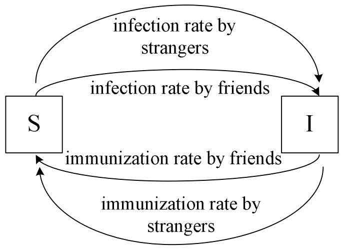

infection rate by strangers
infection rate by friends
S
I
immunization rate by friends
immunization rate by strangers

Fig. 7 Markov chain model of the S-I-S infectious disease with susceptible state and infected state (with finite buffer)

Figure 7 illustrates the Markov chain model of S-I-S. Similar to the infinite buffer case, we can get the infection rate and the immunization rate and have the mass balance equations for $I_i(t)$ and $I_u(t)$:

$$\frac{dI_i}{dt} = I_i S_i(\beta_f \gamma_f + \beta_s \gamma_s) - (N_i \beta_f \alpha_f + N_u \beta_s \alpha_s) I_i$$

$$\frac{dI_u}{dt} = I_u S_u(\beta_f \gamma'_f + \beta_s \gamma'_s) - (N_i \beta_f \alpha_f + N_u \beta_s \alpha_s) I_u$$

With $S_i = N_i - I_i$ and $S_u = N_u - I_u$ we get:

$$\begin{aligned}\frac{dI_i}{dt} &= I_i(N_i - I_i)(\beta_f\gamma_f + \beta_s\gamma_s) - (N_i\beta_f\alpha_f + N_u\beta_s\alpha_s)I_i \\ &= ((\beta_f\gamma_f + \beta_s\gamma_s)N_i - (N_i\beta_f\alpha_f + N_u\beta_s\alpha_s))I_i \\ &\quad -(\beta_f\gamma_f + \beta_s\gamma_s)I_i^2 \\ &= ((\beta_f\gamma_f + \beta_s\gamma_s)N_i - (N_i\beta_f\alpha_f + N_u\beta_s\alpha_s))I_i \\ &\quad \cdot(1 - \frac{I_i}{N_i - \frac{N_i\beta_f\alpha_f + N_u\beta_s\alpha_s}{\beta_f\gamma_f + \beta_s\gamma_s}})\end{aligned} \tag{9}$$

and

$$\begin{aligned}\frac{dI_u}{dt} &= I_u(N_u - I_u)(\beta_f\gamma_f' + \beta_s\gamma_s') - (N_i\beta_f\alpha_f + N_u\beta_s\alpha_s)I_i \\ &= ((\beta_f\gamma_f' + \beta_s\gamma_s')N_u - (N_i\beta_f\alpha_f + N_u\beta_s\alpha_s))I_u \\ &\quad \cdot(1 - \frac{I_u}{N_u - \frac{N_i\beta_f\alpha_f + N_u\beta_s\alpha_s}{\beta_f\gamma_s' + \beta_s\gamma_f'}})\end{aligned} \tag{10}$$

where α_f and α_s are the purging rates of friends and strangers (i.e., the probability that one data will be purged out from the buffer at each contact).

For the logistic differential Eqn. (9) and (10), since $\beta_f\gamma_f + \beta_s\gamma_s$, $N_i\beta_f\alpha_f + N_u\beta_s\alpha_s$, $\beta_f\gamma_f' + \beta_s\gamma_s'$, and $N_i\beta_f\alpha_f + N_u\beta_s\alpha_s$ are larger than 0, as long as $\frac{(\beta_f\gamma_f+\beta_s\gamma_s)N_i}{N_i\beta_f\alpha_f+N_u\beta_s\alpha_s}$ and $\frac{(\beta_f\gamma_f'+\beta_s\gamma_s')N_u}{N_i\beta_f\alpha_f+N_u\beta_s\alpha_s}$ exceeds one, the endemic equilibrium of Eqn. (9) and (10) can be reached when $\frac{dI_i}{dt} = 0$ and $\frac{dI_u}{dt} = 0$, respectively.

Therefore, in these two cases, $1 - \frac{I_i}{N_i - \frac{N_i\beta_f\alpha_f+N_u\beta_s\alpha_s}{\beta_f\gamma_f+\beta_s\gamma_s}}$ is equal to 0 and $1 - \frac{I_u}{N_u - \frac{N_i\beta_f\alpha_f+N_u\beta_s\alpha_s}{\beta_f\gamma_s'+\beta_s\gamma_f'}}$ is equal to 0, which means,

$$I_i = N_i - \frac{N_i\beta_f\alpha_f + N_u\beta_s\alpha_s}{\beta_f\gamma_f + \beta_s\gamma_s} \tag{11}$$

and

$$I_u = N_u - \frac{N_i\beta_f\alpha_f + N_u\beta_s\alpha_s}{\beta_f\gamma_f' + \beta_s\gamma_s'} \tag{12}$$

Table 2 shows some numerical results based on our analysis. The results indicate that FdSs and FdSd tend to diffuse and buffer data among nodes that are not interested. Therefore, nodes use more buffer space to hold uninteresting data. However, in FsSd and FsSs, as data has high priority to be diffused between interested nodes, most data copies are at the interested nodes. FsSd differs from FsSs in that it also has high probability to diffuse one particular data item between any two strangers, which brings in more data copies at its uninterested nodes. Since most queries are initiated by the interested nodes and friends are easier to meet with each other, as reported in Table 2. Even though FsSd results in fewer data copies than FdSs and

Table 2 Numerical results of data distribution based on the S-I-S analysis model ($N_i = 20$, $N_u = 80$)

	$\alpha_f/\alpha_s = 0.1/0.9$			$\alpha_f/\alpha_s = 0.2/0.8$			$\alpha_f/\alpha_s = 0.3/0.7$			$\alpha_f/\alpha_s = 0.4/0.6$			$\alpha_f/\alpha_s = 0.5/0.5$		
	I_i	I_u	Delay	I_i	I_u	Delay	I_i	I_u	Delay	I_i	I_u	Delay	I_i	I_u	Delay
FsSd	19	48	4.63	17	52	5.09	14	57	5.57	11	62	6.07	7	67	6.27
FdSs	0	75	5.60	0	73	5.75	0	70	6.00	0	68	6.18	7	67	6.27
FsSs	18	0	10.00	15	27	6.07	12	49	7.07	10	60	6.60	7	67	6.27
FdSd	2	79	5.32	3	77	5.46	5	74	5.68	6	71	5.92	7	67	6.27

FdSd, it can still serve queries faster. Since FsSd has more data copies at the uninterested nodes than FsSs, it can serve the query from uninterested nodes more quickly. We can also observe that as the purging rate becomes skewer, the advantage of FsSd becomes more obvious. This is because when the purging rates become unequal between interested nodes and uninterested nodes, most queries can be served quickly by the interested nodes according to Eqn. (8),(11), and (12).

5 Performance Evaluations

In this section we evaluate the proposed diffusion schemes with real traces. We first study the infinite buffer case and then study the finite buffer case.

5.1 Experiment Setup

To evaluate different diffusion schemes, we use two well-known traces: the Cambridge Haggle Trace [11] and the MIT Reality Mining Trace [12]. In the Cambridge trace, 41 Intel iMotes were distributed to students attending the Infocom student workshop in Miami, 2005. They collected information such as when they meet with each other or any other external new devices. The trace covers 3 days. The MIT trace consists of 100 users carrying Nokia 6600 smart phones over more than nine months. The details of the two experimental traces are briefly summarized in Table 3. We extract the contact information from both traces to identify direct contacts between nodes where data diffusion could have taken place. The trace files are divided into discrete sequential contact events which are fed into our experiments. Each time a contact is observed, the node makes a diffusion decision based on the diffusion scheme.

The interest distribution of each node is generated based on its contact rate with other nodes. We assure that each pair of nodes share more common interests if they have higher contact frequency. Due to the randomness of node activity, in each experiment we characterize the contact rate of each node pair with the first half of the trace. We count the contact times of each node pair and adjust their interest distributions so that two nodes can have more common interests when they meet more of-

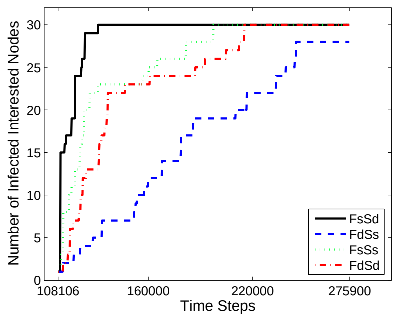
Table 3 Characteristics of the two experimental traces

Experimental trace	*Cambridge* Infocom'05	*MIT* Reality
Device	iMotes	Smart Phones
Network type	Bluetooth	Bluetooth
Duration	3/7/2005∼3/10/2005 (3 days)	7/10/2004∼5/5/2005 (9 months)
Granularity	120 sec	300 sec
# of devices	41 internal, 233 external	97 internal
# of contacts	28,216	113,902

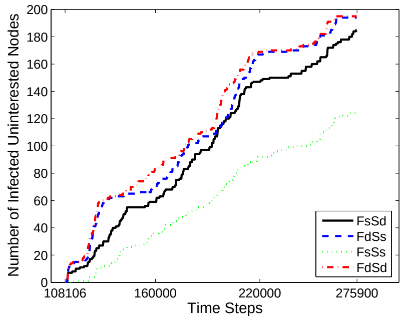
ten. After that we use the second half of the trace to evaluate the proposed schemes. More specifically, in the Cambridge trace, we use the trace from time 21703 to time 108098 as the training dataset and run the experiments with the trace from time 108106. In the MIT trace, the first half of the trace (from 2004/7/10, 15:57 to 2004/11/19, 11:52) is used to predict node relationship and the remainders are fed into the diffusion schemes. We generate 1000 queries which are uniformly distributed in the whole experiment period. Following Pareto's Rule [35], 80% queries are initiated by interested nodes and other 20% are initiated by uninterested nodes. We also divide the whole experiment period into n ($n = 15$ in *Cambridge* trace and $n = 14$ in *MIT* trace) statistical sessions to record the query delay results. We use the average delay of all queries in each session to represent the query delay of that statistical session. In order to overcome the finiteness of the traces, if the initiated query is not served before the end of the trace, the same meeting pattern is applied by re-feeding the trace from the beginning. Each experiment is repeated 10 times with different random seeds to eliminate randomness.

5.2 *The Infinite Buffer Case*

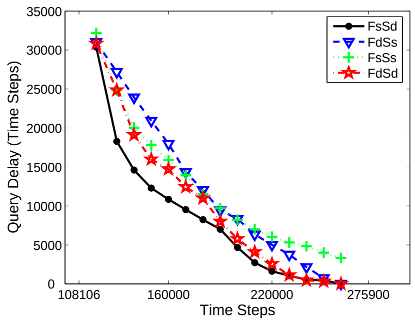
Figure 8 compares the data diffusion speed and data access delay as a function of time for the four schemes. As shown in the figure, with infinite buffer, the number of infected nodes increases and the query delay decreases as time goes. However, the diffusion speed and query delay under different scheme is different. As shown in Figure 8(a), data can be diffused to its interested nodes more quickly in FsSd and FsSs than that in FdSs and FdSd. This result is consistent with our numerical analysis in which FsSd and FsSs have faster diffusion speed among interested nodes. Since FsSd and FsSs assign high diffusion priority among interested nodes and nodes sharing similar interests are more likely to meet with each other, the data can easily spread out among its interested nodes. Instead, FdSs and FdSd diffuse data slowly among interested nodes but they have a faster diffusion speed among uninterested nodes as shown in Figure 8(b). This is because friends diffuse different data first in FdSs and FdSd, and the data has more opportunity to be diffused between two uninterested friends. Further, FdSd can speed up data diffusion be-



(a) Number of Infected Interested Nodes



(b) Number of Infected Uninterested Nodes



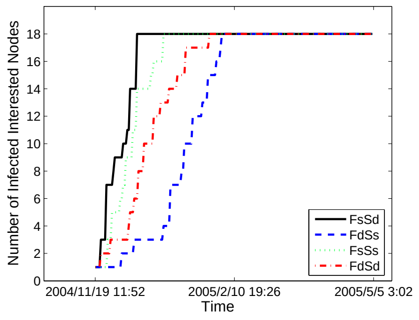
(c) Query Delay

Fig. 8 Results for the Cambridge Infocom'05 trace (with infinite buffer)

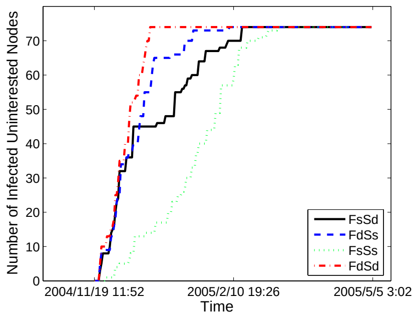
tween two strangers who are not interested in the data because two strangers diffuse different data first in FsSd.

As shown in Figure 8(c), as the number of data copies increases, queries for these data can be served more quickly. Since FsSd diffuses data faster among interested nodes, it has the shortest query delay compared with other schemes. For example, at time 160000, the query delay of FsSd is about 26% less than FdSd, 32% less than FsSs, and about 40% less than FdSs. At time 220000, the performance difference is more obvious. The query delay of FsSd is about 37% and 73% less than FdSd and FsSs, respectively, and 68% less than FdSs. Note that FsSs has the longest delay because more than 100 uninterested nodes are still uninfected in FsSs when the experiment finishes.

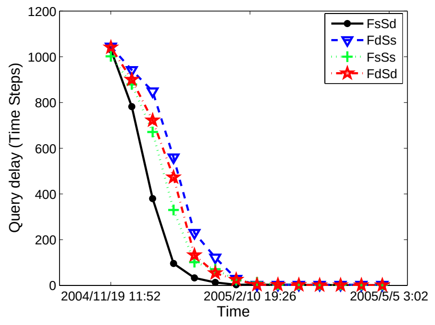
Figure 9 presents comparison results based on the MIT Reality trace. Again, FsSd achieves the best performance in terms of data diffusion speed and query delay. It has the fastest diffusion speed among interested nodes and the shortest query delay, which is consistent with the results of the Cambridge trace, and the analytical results in the last section. We notice that the diffusion speed is much slower in



(a) Number of Infected Interested Nodes



(b) Number of Infected Uninterested Nodes



(c) Query Delay

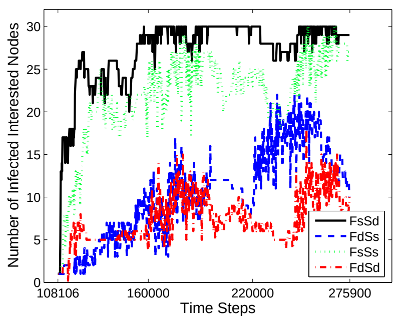
Fig. 9 Results for the MIT Reality Mining trace (with infinite buffer)

the Cambridge trace than that in the MIT trace. This is because most nodes in the Cambridge trace are external nodes, which do not appear regularly in the network.

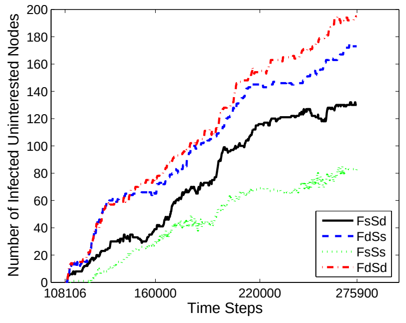
5.3 *The Finite Buffer Case*

With finite buffer, some data may be replaced if the buffer is full. As shown in Figure 10(a) and Figure 10(b), the number of infected interested nodes and the number of infected uninterested nodes fluctuate at different time. This fluctuation comes from the fact that the data can be diffused among nodes and can be removed from the buffer as well. When the data item is buffered, the number of infected nodes increases. If the node's buffer is full, some data item has to be removed. Then, the node returns to the susceptible status and the number of infected nodes decreases.

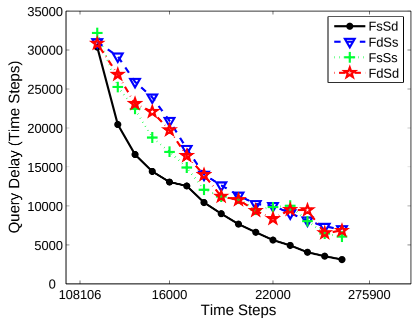
In FsSd, each node prefers buffering its interesting data rather than uninteresting data. As long as a data item is buffered at its interested node, it will not be removed most likely. Thus, there are always more infected interested nodes in FsSd. FdSs



(a) Number of Infected Interested Nodes



(b) Number of Infected Uninterested Nodes



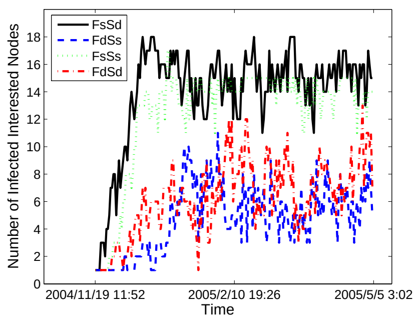
(c) Query Delay

Fig. 10 Results for the Cambridge Infocom'05 trace (with finite buffer)

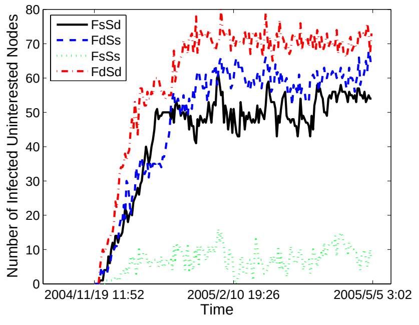
and FdSd are different. They diffuse different data and remove similar data first between friends. Consequently, there will not be many data copies at the interested nodes. However, FdSs and FdSd give high priorities to diffuse and buffer data in the uninterested nodes. Hence, they have more data copies in uninterested node than FsSd and FsSs. Even though there are fewer infected uninterested nodes in FsSd, FsSd still outperforms FdSs and FdSd in terms of query delay because it helps diffuse data to the interested nodes. As shown in Figure 10(c), FsSd can reduce up to 60% query delay compared to the other three schemes.

Figure 11 shows comparisons based on the MIT trace. The results are similar to that of the Cambridge trace. Because the MIT trace logs fewer nodes, but with more activities, the prediction on the contact rate with the first half of trace data is more accurate, and thus making FsSd perform better.

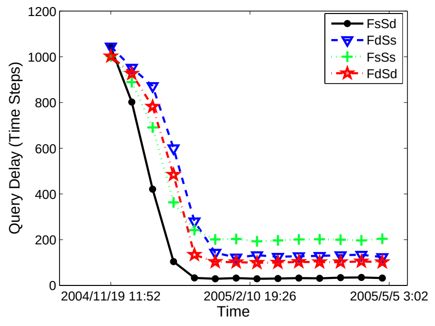
By comparing Figure 8(c) to Figure 10(c), we can see that the query delay in the infinite buffer case is much shorter than that in the finite buffer case, and the difference becomes more obvious as time goes. Similar results can be seen by comparing Figure 9(c) and Figure 11. This is because data may be purged out when the buffer is full in the finite buffer case. As a result, the delay for the finite buffer case is longer



(a) Number of Infected Interested Nodes



(b) Number of Infected Uninterested Nodes



(c) Query Delay

Fig. 11 Results for the MIT Reality Mining trace (with finite buffer)

than that in the infinite buffer case. Similarly, the data fusion speed is also higher in the infinite buffer case than that in the finite buffer case.

5.4 Discussion

It is worth noticing that although FsSd has the best performance among the four schemes, its diffusion probability between two uninterested friends is still low, which slows down the diffusion speed among uninterested nodes. This is because in FsSd, friends first choose the data that is more similar to their common interests to diffuse, which prevents the diffusion of their uninteresting data (γ_f' is set to a small value in Table 1). To diffuse one data item quickly between uninterested friends, some changes have to be made in FsSd. For example, each pair of friends have to make different diffusion decisions on their interesting data and uninteresting data, i.e., to diffuse similar data first for the interesting data and to diffuse different data first for uninteresting data.

However, this modified scheme may not be practical, because it is impossible to tell whether the interesting data or uninteresting data is more important and a node cannot treat interesting data and uninteresting data separately. Further, according to the modified scheme, all data items have the same diffusion priority. Then, suppose one data item could have the diffusion privilege at all nodes, all data will have high diffusion priority, which is also impossible in a competition system. Although FsSd does not have the fastest diffusion speed among all nodes, it can diffuse data to the interested nodes quickly, which helps reduce the overall query delay.

6 Conclusions

In this chapter, we studied the performance of different data diffusion schemes in delay tolerant mobile ad hoc networks (DT-MANETs). We introduced four possible social-aware data diffusion schemes and developed theoretical models to analyze their performance in terms of data diffusion speed and query delay. Based on the analysis, we found an interesting result: to achieve better performance, a node should first diffuse the data most similar to their common interest when it meets a friend, and it should first diffuse the data most different to their common interest when it meets a stranger. To verify the theoretical result, extensive experiments have been carried out based on real traces of human contacts, and the experimental results are consistent with our theoretical analysis.

To the best of our knowledge, our work is the first to study data diffusion instead of data forwarding/routing using sociological knowledge. In this initial effort, of course, we have not addressed all relevant problems. In the future, we will look into other techniques to measure interest similarity. We will also investigate how to integrate data forwarding and data diffusion.

Acknowledgments

This work was supported in part by Network Science CTA under grant W911NF-09-2-0053.

References

1. S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. In *SIGCOMM*, pages 145–158, 2004.
2. W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc*, pages 187–198, 2004.
3. 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). In *MobiHoc*, pages 233–244, 2003.

4. A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, 2007.
5. E. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *MobiHoc*, pages 32–40, 2007.
6. P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: Social based forwarding in delay tolerant networks. In *MobiHoc*, 2008.
7. P. Costa, C. Mascolo, M. Musolesi, and G.P. Picco. Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 26(5):748–760, June 2008.
8. J. Ghosh, S. J. Philip, and C. Qiao. Sociological orbit aware location approximation and routing (solar) in manet. *Ad Hoc Netw.*, 5(2):189–209, 2007.
9. J. Burgess, B. Gallagher, D. Jensen, and B.N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *INFOCOM*, 2006.
10. M. McPherson, L. Smith-Lovin, , and J. Cook. Birds of a feather: Homophily in social networks. In *Annual Review of Sociology*, pages 15–44, 2001.
11. Cambridge Haggle Project. http://www.haggleproject.org/.
12. MIT Realisty Mining Project. http://reality.media.mit.edu/.
13. M. Motani, V. Srinivasan, and P. S. Nuggehalli. Peoplenet: engineering a wireless virtual social network. In *MobiCom*, pages 243–257, 2005.
14. W. Gao, Q. Li, B. Zhao, and G.Cao. Multicasting in delay tolerant networks: A social network perspective. In *MobiHoc*, 2009.
15. F. Bai and A. Helmy. Impact of mobility on last encounter routing protocols. *SECON*, pages 461–470, June 2007.
16. W. Gao, and G. Cao. On Exploiting Transient Contact Patterns for Data Forwarding in Delay Tolerant Networks. In *IEEE International conference on network protocols(ICNP)*, 2010.
17. Q. Li, S. Zhu, and G. Cao. Routing in socially selfish delay tolerant networks. In *INFOCOM*, 2010.
18. Y. Zhang, J. Zhao, G. Cao, and C. Das. On interest locality in content-based routing for large-scale manets. In *IEEE 6th International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pages 178–187, 2009.
19. A. Miklas, K. Gollu, K. Chan, S. Saroiu, K. Gummadi, and E. Lara. Exploiting social interactions in mobile systems. In *UbiComp*, 2007.
20. T. Karagiannis, J. Boudec, and M. Vojnović. Power law and exponential decay of inter contact times between mobile devices. In *MobiCom*, pages 183–194, 2007.
21. Y. Wang, B. Krishnarnachari, and T. Valente. Findings from an empirical study of fine-grained human social contacts. In *The Sixth International Conference on Wireless On-Demand Network Systems and Services (WONS)*, pages 141–148, 2009.
22. W. Hsu, D. Dutta, and A. Helmy. Mining behavioral groups in large wireless lans. In *MobiCom*, pages 338–341, 2007.
23. J. Zhao and G. Cao. VADD: vehicle-assisted data delivery in vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, 57(3):1910–1922, May 2008.
24. Y. Zhang, J. Zhao, and G. Cao. Roadcast: a popularity aware content sharing scheme in vanets. In *IEEE ICDCS*, pages 223–230, 2009.
25. S. Kapadia, B. Krishnamachari, and S. Ghandeharizadeh. Static replication strategies for content availability in vehicular ad-hoc networks. *Journal of Mobile Network and Applications (MONET)*, 14(5):590–610, 2009.
26. S. Ghandeharizadeh and S. Kapadia. An evaluation of location-demographic replacement policies for zebroids. In *IEEE Consumer Communications and Networking Conference (CCNC)*, Jan 2006.
27. C. Boldrini, M. Conti, and A. Passarella. Contentplace: social-aware data dissemination in opportunistic networks. In *MSWiM*, pages 203–210, 2008.
28. V. Lenders, G. Karlsson, and M. May. Wireless ad hoc podcasting. In *SECON*, pages 273–283, 2007.

29. X. Zhang, G. Neglia, J. Kurose, and D. Towsley. Performance modeling of epidemic routing. *Comput. Netw.*, 51(10):2867–2891, 2007.
30. K. Lee, M. Le, J. Haerri, and M. Gerla. Louvre: Landmark overlays for urban vehicular routing environments. *IEEE WiVeC*, 2008.
31. Q. Yuan, I. Cardei, and J. Wu. Predict and relay: an efficient routing in disruption-tolerant networks. In *MobiHoc*, pages 95–104, 2009.
32. RDF Core Working Group http://www.w3.org/RDF/.
33. Web Services Description Language (WSDL) Version 2.0 http://www.w3.org/TR/wsdl20.
34. S. Kullback and R. A. Leibler. On information and sufficiency. In *Annals of Mathematical Statistics 22: 79-86.*, 1951.
35. W. J. Reed. The pareto, zipf and other power laws. In *Economics Letters*, pages 15–19, 2001.