Author Name

Social Network Analysis: Interdisciplinary Approaches and Case Studies

List of Tables

Chapter 1

Information dissemination in social-featured opportunistic networks

Wenzhong Li, Sanglu Lu

State Key Laboratory for Novel Software Technology, Department of Computer Science and Technology, Nanjing University, 210023, China

Konglin Zhu

School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, 100876, Beijing, China

Xiao Chen

Department of Computer Science, Texas State University, San Marcos, TX USA

Jan Nagler

MPI for Dynamic and Self-organization, Goettingen, Germany
ETH Zrich, Switzerland

Xiaoming Fu

Institute of Computer Science, University of Goettingen, D-37077, Goettingen, Germany

CONTENTS

1.1 Introduction

In recent years, opportunistic networks have emerged as a mechanism for infrastructure-free communications in wireless networks. An opportunistic network is a sparse dynamic wireless network where mobile nodes work in ad hoc mode and can communicate with each other only when they move into their communication range [8]. The communication in opportunistic networks is disruption-tolerant, and there is no need of establishing end-to-end message routing paths. Since opportunistic networks allow people to communicate without network infrastructure, they are widely used in wildlife tracking sensor networks, vehicular ad hoc networks, pocket switched networks, and mobile social networks.

Information dissemination addresses the issue of sending a piece of information from a source node to one or more destination(s), which is a key function of communication in opportunistic networks. Since mobile devices can exchange information only when humans come into contact, such networks are tightly coupled with human social networks [24]. Therefore, information dissemination in opportunistic networks can exploit social features such as users' social profiles, social relationships and network structures to build more efficient dissemination schemes.

In this chapter, we introduce two types of information dissemination in social-featured opportunistic networks: *unicast* and *multicast*.

Unicast enables one-to-one communication in opportunistic networks, where one mobile node sends information to another node. Based on characterizing users' social interactions and mobility patterns, we propose a ***S***ocial- and ***M***obile-***A***ware message ***R***ou***T***ing strategy called ***SMART***. It exploits a distributed community partitioning algorithm to divide an opportunistic network into smaller communities based on user movements and interaction routines. Then according to the positions of the destination nodes, it either performs the intra-community communication or the inter-community communication process to disseminate information. For intra-community communications, a decayed routing utility combining social similarity and social centrality is calculated, which is used to decide relay nodes efficiently inside the community. To enable efficient inter-community communications, it chooses the fringe nodes that travel remotely as relays, and the community-degree utilities are calculated for routing decision across communities. The efficiency of SMART is e-valuated by extensive trace-driven experiments which illustrate that it outperforms other unicast strategies in various opportunistic network traces.

Multicast enables one-to-many communication where one mobile node send-s information to a set of destinations. We introduce the concept of dynamic social features and its enhancement to capture nodes' dynamic contact behavior and social relationships. Based on the derived dynamic social features and the community structure, we adopt the compare-split scheme to select the best relay node for each destination in each hop to construct a multicast tree. Specifically, we propose two ***C***ommunity and ***S***ocial feature-based multicast algorithms named *Multi-CSDO* that involves ***D***estination nodes ***O***nly in community detection and *Multi-CSDR* that involves both the ***D***estination nodes and the ***R***elay candidates in community detection in case the relay candidates are also socially similar. The performance of the algorithms is evaluated by simulations with a real trace of an mobile social network, which shows that the proposed multicast algorithms outperform the existing ones in terms of delivery rate, latency, and number of forwardings.

The rest of the chapter is organized as follows. Section 1.2 presents the modeling of opportunistic network as a social graph. Section 1.3 introduces the definitions of social features and social properties. Section 1.4 proposes the information dissemination strategies for unicast. Section 1.5 proposes the information dissemination s-trategies for multicast in opportunistic networks. The chapter is concluded in section 1.6.

1.2 Model

We model an opportunistic network as a social graph $G = < V, E >$, which is illustrated in Fig. 1.1. In the graph, V is a set of nodes representing mobile users. Each

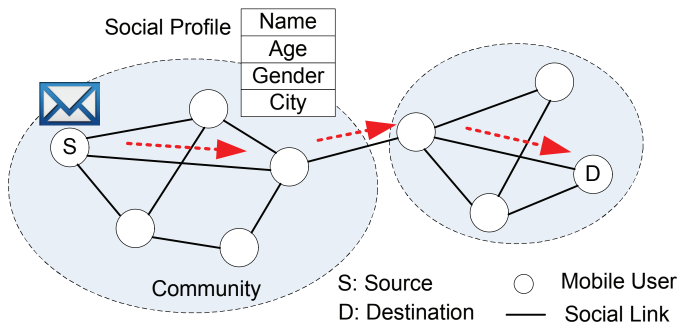


Figure 1.1: The social graph of an opportunistic network.

user has a set of personal information forming his social profile. The *encounter event* is defined as the event that two nodes enter the communication range of each other. If a node encountered another node in the past, there is a link between them indicating their social relationship. E is the set of links/edges in the graph.

The following glossary is used to describe the social properties of social-featured opportunistic networks.

Social ties: represent the social interaction between two nodes, which is quantified by the frequency of their encounters.

Neighbors: represent the set of nodes that have direct contact to a user in the network.

Communities: represent the clustering structure of the social network. Nodes within the same community are closely connected to each other either by direct linkage or intermediates.

Social features of opportunistic network is defined in the next section.

1.3 Social features

Since information diffusion in opportunistic networks is based on the contacts of mobile devices carried by human individuals, the speed and range of diffusion are dependent upon the frequency and the patterns of intercontacts of individuals, which are affected by the social features of individuals. In this section, we introduce the representation of social features in opportunistic networks. Specifically, we explore two types of social features: social profiles and social network structures.

1.3.1 Social Profiles

Social profiles are the properties associated with individuals in the social life. For example, node (an individual) i may be a *female* working as a *faculty* member in a *university in USA*, while node j may be a *male* working as a *manager* in a *company in*

China. The two nodes have different gender, profession, affliction and and locations that form their social profiles. According to the study of [26, 27], people come in contact more frequently if they have more social profile features in common.

We use a vector $E_i =< E_{i1}, E_{i2}, \cdots, E_{im} >$ to indicate the social profiles of node i, where E_{i1}, E_{i2}, $\cdots$ can refer to the information of *nationality*, *city*, *language*, *gender*, *profession*, *affliction*, etc. Each social feature E_{ik} can take multiple values. For example, a social feature E_{ik} can be Language and its values can be English, Spanish, and so on. The social profile of a person is time variant. For example, the *age* and *location* of a person will change over time. Thus we use $E_{ik}(\tau)$ to indicate the value of the kth social feature of node i at time τ. We introduce the *social profile similarity* to represent the similarity of two individuals regarding their social profiles.

Definition 1.1 Social profile similarity Assume $E_i(\tau) =< E_{i1}(\tau), E_{i2}(\tau), \cdots, E_{im}(\tau) >$ and $E_j(\tau) =< E_{j1}(\tau), E_{j2}(\tau), \cdots, E_{jk}(\tau) >$ are the social profiles of node i and j at time τ, the social profile similarity of the two nodes are defined as

$$S_{i,j}(\tau) = dist(E_i(\tau), E_j(\tau)), \tag{1.1}$$

where $dist(X,Y)$ are the distance measurement of the two vectors.

Based on the definition, the social profile similarity can be measured by the distance of two vectors. There are a variety of distance measurements such as Euclidean distance, Mahalanobis distance, etc., which can be used to quantify the social profile similarity.

Another metric to evaluate the closeness of two nodes is *social neighbor similarity*, which is defined as the number of neighbors they shared.

Definition 1.2 Social structure similarity Assume $\mathcal{F}_i(\tau)$ and $\mathcal{F}_j(\tau)$ are the sets of neighbors of nodes n_i and n_j at time τ accordingly, the social neighbor similarity of the two nodes is defined as

$$S_{i,j}(\tau) = |\mathcal{F}_i(\tau) \cap \mathcal{F}_j(\tau)|. \tag{1.2}$$

Social neighbor similarity infers the reachability of two nodes for information dissemination in opportunistic networks. The higher the social neighbor similarity of two nodes is, the more likely they can communicate with each other via the common neighbors.

1.3.2 Social network structures

Social network structure represents the position of a node in the social network and how the nodes connect with each other in the network. We use *centrality* and *community* to specify such social features.

Social centrality refers to the position and the relative importance of a node in the social network structure. There are various definitions of centrality which includes degree centrality [11] and betweenness centrality [10].

Definition 1.3 Degree Centrality Degree centrality is simply defined as the proportion of the number of links incident upon a node. Assume $G =< V, E >$ is a social graph; d_{ik} is an indicator to represent the existence of a link between two nodes, where $d_{ik}(\tau) = 1$ if there exists a link between node i and node k at time τ, and $d_{ik}(\tau) = 1$ otherwise. The degree centrality of node i at time τ is calculated by

$$C_i(\tau) = \frac{\sum_{\forall k \in V} d_{ik}(\tau)}{\sum_{\forall j \in V} \sum_{\forall k \in V} d_{jk}(\tau)}, \tag{1.3}$$

where $\sum_{\forall j \in V} \sum_{\forall k \in V} d_{jk}(\tau)$ is the total number of links in the social network.

Definition 1.4 Betweenness Centrality Betweenness centrality measures the importance of a node that acts as a bridge along the shortest path between two other nodes in the social graph. For a social graph $G =< V, E >$, assume at time τ, the set of nodes on the shortest path from node s to node t is indicated by $V_{s \to t}$. The betweenness centrality of node i at time τ is calculated by

$$C_i(\tau) = \frac{\sum_{\forall s \in V \& s \neq i} \sum_{\forall t \in V \& t \neq i} |\{i\} \cap V_{s \to t}|}{(n-1)(n-2)/2}, \tag{1.4}$$

where the upper part calculates the total times that node i lies on the shortest path from s to t, and the lower part is the total number of source-destination pairs in the social graph.

In the context of information diffusion in social networks, the degree centrality and betweenness centrality can be interpreted in terms of the ability of a node for holding information flowing through the network.

The community structure refers to the property that nodes of the network forming subgroups with which vertex-vertex connections are dense, but between which connections are less dense [12]. According to [21], the community structure of a social network can be defined as follows.

Definition 1.5 Community A community is a local densely connected subgraph in a network. For a subgraph $G' =< V', E' >$ of a social graph $G =< V, E >$, assume d_i^{in} is the number of connections from node i to the other nodes in V', and d_i^{out} is the number of connections from i to the nodes in $V - V'$. G' is a community of G if it satisfies

$$\sum_{i \in V'} d_i^{in} > \sum_{i \in V'} d_i^{out}, \tag{1.5}$$

i.e., the total number of internal connections in G' should be larger than the total number of external connections to the rest of the network.

Community detection algorithms have been well studied in the past. Typical approaches include the min-cut technique [6] that partitions a connected graph into subgraphs recursively, the Girvan-Newman algorithm [12] that removes the edges with the highest edge betweenness gradually, and the label propagation algorithm [20] that provides a near-linear time solution for community detection in large-scale networks.

1.4 Unicast

Unicast refers to one-to-one communication in a network. When a piece of information is sent by a source node, a unique destination address is specified. Unicast in opportunistic networks employs the "store-carry-forward" manner: when information is sent by a source node, it traverses several intermediate nodes which store and carry data while moving and forward the data to the next relay node upon encountering, until eventually the destination is reached.

To achieve unicast information delivery in opportunistic networks based on user contacts, we propose a ***S***ocial- and ***M***obile-***A***ware message ***R***ou***T***ing strategy called ***SMART***. The basic idea of SMART is to exploit community structure in opportunistic networks and choose relay nodes to route data adaptively according to social features and mobility characteristics. The SMART scheme first applies a *distributed community partitioning* algorithm to divide mobile nodes into communities, and then based on the positions of the destination nodes, it either performs an *intra-community communication* or an *inter-community communication* process to disseminate information. The details are presented as follows.

1.4.1 Distributed Community Partitioning

In opportunistic networks, communities are formed based on the locations and movement trajectories of users. Intuitively, people staying in closer geographic areas or sharing similar location interests tend to meet each other more often [9, 16, 17]. According to the observation in [28], about 80% of trajectory coordinates of a user appear within 5km from its *centroid* (or known as *geographic mass point*), and the encountering probability of two nodes is high only when their mass points are close enough. Inspired by the previous observation, we propose a dynamic and distributed community partitioning algorithm called *m-partition* to detect the clusters of frequently encountering nodes.

Given the number of community *m*, the community partitioning process contains two stages: the bootstrap stage and the evolution stage. In the *bootstrap stage*, *m* nodes are randomly selected and each node is assigned with a unique community ID. A node without community affiliation will be assigned the ID of the node with know community it first encountered. After this stage, each node has an initial community ID. In the *evolution stage*, each node keeps counting the affiliation parameters (APs), which indicate the number of encounters with nodes in different communities. Then it adjusts its community affiliation according to the updated AP values. Specifically,

Algorithm: m-partition

```
Require: Node $N_i$ and AP vector;
Ensure: The community ID of node $N_i$;
 1: Assume there are $m$ communities to be detected;
 2: for Each encounter event (with $N_j$) do
 3:     if $N_i.communityID = null$ then
 4:         $N_i.communityID = N_j.communityID$
 5:     else
 6:         $y \leftarrow N_j.communityID$
 7:         $x \leftarrow N_i.communityID$
 8:         if $y = x$ then
 9:             $ap_{x_i} = ap_{x_i} + 1$
10:         else
11:             $ap_{y_i} = ap_{y_i} + 1$;
12:         end if
13:         if $ap_{y_i} > ap_{x_i}$ then
14:             $N_i.communityID = y$
15:         end if
16:     end if
17: end for
```

Figure 1.2: The m-partition algorithm.

we use a vector to depict the affiliation parameters of node n_i:

$$E_i = \{ap_{1_i}, ap_{2_i}, \cdots, ap_{m_i}\},$$

where ap_{j_i} is the AP of n_i connecting to community C_j, which denotes the number of encounters between n_i and nodes in C_j. The AP vector is updated each time an encounter occurs, and the node adaptively changes its community affiliation to the community with the maximal AP value in the vector. The detailed process of the algorithm is illustrated in Fig. 1.2.

The algorithm is run in a distributed way, and each node keeps recording the community affiliations of other nodes it encounters. Since community is formed by a group of frequently encountering nodes, the community members will be known to each other after running the algorithm long enough.

After dividing mobile users into communities, we consider two cases of unicast: the *intra-community communication* where the source and destination are in the same community, and the *inter-community communication* where the source and destination are in different communities. The principles of information dissemination for the two cases are discussed in the following sections.

1.4.2 Intra-Community Communication

In the social graph of an opportunistic network, the links between nodes indicate the encountering events of the nodes in the past. If a source node n_s and a destination node n_d are in the same community, they will encounter each other or other nodes in the same community more often, thus the structure of the social graph can be exploited to form a routing path between n_s and n_d. Intuitively, we can choose relay nodes based on social features such as centrality and similarity. On one hand, if a node has more links to the other nodes, it is more likely to encounter the destination, so social centrality is a good indicator of the ability to serve as a hub for information exchange. On the other hand, if a node has more common friends with the destination node, it will have higher probability to reach the destination directly or indirectly (via a common friend), thus social neighbor similarity can also be used as a metric to choose relay nodes. In this section, we will combine the two metrics to form a utility function for routing decision.

According to section 1.3.2, social centrality can be represented by the degree centrality of a node. However, to multigate the accumulative effect that a node may encounter many other nodes in the history but become less active in the recent, we introduce a decayed degree centrality of node i to overcome accumulative effect of historical encounter events, which is calculated by

$$C'_i(\tau) = \frac{C_i(t)}{\tau - t}, \tag{1.6}$$

where τ is the current time and t is the most recent time that node i encounters another node. According to the equation, $C'_i(\tau)$ is a decay function of $C_i(t)$.

Social neighbor similarity is given by equation (1.2). We calculate the decayed social neighbor similarity as

$$S'_{i,j}(\tau) = \frac{S_{i,j}(t)}{\tau - t}, \tag{1.7}$$

where τ is the current time and t is the most recent time that node i encounters j or j's friends.

Since social centrality and social structure similarity describe different aspects of social features of a mobile node, there need to be a way to combine the two different metrics to form a utility function. Inspired by the concept of convolution in signal processing, which provides a mathematical operation on two functions to produce the weighted average over time, the utility function at time T is formulated as

$$Y_{i,d}(T) = S'_{i,d}(T) \otimes C'_i(T) = \int_{\tau=0}^{T} S'_{i,d}(\tau) * C'_i(T-\tau). \tag{1.8}$$

In the real implementation, time is divided into slots, and the utility can be calculated in a discrete way and can be updated whenever an encountering event occurs:

$$U_{i,d}(T) = \sum_{\tau=0}^{T} X_{i,d}(\tau) * S'_{i,d}(\tau) * C'_i(T-\tau), \tag{1.9}$$

where $X_{i,d}(\tau) = 1$ when an encounter occurs at time τ; otherwise, $X(\tau)_{id} = 0$.

The utility function describes that when each encounter occurs, it yields an addictive effect represented by social structure similarity and social centrality decaying over time. According to the utility function, a node with higher degree and more common friends with the destination decays slower than a node with poor connection to the network, and a node with more recent encounters decays slower than a less active one.

With the derived utility function, we propose the routing principle for intra-community communication.

Intra-community forwarding principle: In opportunistic networks, if a source n-ode sends a message to a destination within the same community, whenever an intermittent node n_i is encountered, utility function in Eq. (1.9) is applied, and message is forwarded to the node with higher utility until the destination is reached.

1.4.3 Inter-Community Communication

If the destination node n_d does not belong to the same community as the source node n_s, we need to choose some relay nodes to forward the message across communities. The idea is to use "fringe nodes" to bridge the communication of inter-communities.

Table 1.1: Remote contact table of node n_i

$\mathcal{X}_1$	$\mathcal{X}_2$	$\cdots$	$\mathcal{X}_k$	$\cdots$	$\mathcal{X}_M$
η_{i1}	η_{i2}	$\cdots$	0	$\cdots$	η_{iM}

A fringe node is a node which is capable of remote contact with other communities. It is measured by the number of links that it connects to other communities. We select nodes with higher number of links to outside communities as fringe nodes. Each fringe node is represented by its ID and its remote contact table is shown in Table 1.1, where $\mathcal{X}_k$ $(k = 1, \cdots, M)$ is the community ID, and η_{ik} $(k = 1, \cdots, M)$ is the frequency that n_i encounters nodes in $\mathcal{X}_k$.

Each community maintains a set of fringe nodes $\mathcal{F}$. The set $\mathcal{F}$ is calculated and updated periodically. During a period, each node compares the contact frequencies in its remote contact table with those of the fringe nodes in $\mathcal{F}$. If a node n_i finds that it has better connection with outside communities than a fringe node n_j, it will announce itself as a new fringe node by broadcasting its ID and remote contact table to the community. The process is described below.

Assume η_{i1}, η_{i2}, $\cdots$, η_{iM} are the remote contact frequencies of n_i, and η_{j1}, η_{j2}, $\cdots$, η_{jM} are the remote contact frequencies of n_j. Define a function $\phi(x,y) = 1$ if $x \geq y$; and otherwise $\phi(x,y) = -1$. The relative ability for remote contact of the two nodes is evaluated by

$$\mathfrak{F}_{i,j} = \sum_{k=1}^{M} \phi(\eta_{ik}, \eta_{jk}).$$

If $\mathfrak{F}_{i,j}$ is larger than 1, it means n_i has better remote connection than n_j, thus n_i will announce itself as a fringe node for the community.

According to the report in [25], a small fraction of the remote links are enough to form a small world network with a small network diameter. In our network, we set the number of fringe nodes as 10% of the community size. If the community size is smaller than 10, the number of fringe nodes is set as 1.

Inter-community forwarding principle: For inter-community communication, a source node n_s in community C sends a message to the destination n_d in community C', where $C \neq C'$. With the set of fringe nodes derived, it applies the following principles to forward the message from C to C'.

(1) If C and C' are directly connected, i.e., there exists a non-empty set $\mathbb{C} = \{n_j | \forall\ n_j \in \mathcal{F}\ and\ n_j\ connects\ to\ C'\}$, the principle is to choose the fringe node with the maximum connection to C' as relay. That is, the message will be forwarded to the fringe node with higher degree centrality to C'.

(2) If C and C' are not directly connected, the message will be forwarded across multiple communities. Similar to the definition of the degree centrality, we define the *community-degree centrality* as follows.

$$\mathcal{D}_i = \frac{\sum_{k=1}^{M} \eta_{ik}}{\sum_{\forall j \in F} \sum_{k=1}^{M} \eta_{jk}}. \tag{1.10}$$

A higher $\mathcal{D}_i$ value indicates more connections from n_i to the outside communities. Thus the principle for cross community communication is to forward the message to the fringe node with a higher community-degree centrality. The message will be forwarded in multiple hops, until it reaches the destination community. After that, intra-community forwarding principle will be applied to keep forwarding the message to the destination eventually.

1.4.4 Performance Evaluation

1.4.4.1 Datasets

Our study is based on three publicly available opportunistic network traces: MIT Reality [7], DieselNet [3] and Cabspotting [19]. The MIT Realitydata set consists of the location traces of 97 users with Nokia 6600 smart phones at MIT during the 2004-2005 academic year. DieselNet logs mobility traces of 34 buses in Amherst. Each bus is equipped with a computer and a GPS. It records the GPS locations of all the buses during the 20 days from October to November in 2007. Cabspotting is a mobility trace of taxi cabs in San Francisco. Each taxi is outfitted with a GPS tracking device. It contains GPS coordinates of 536 taxis collected over 30 days in San Francisco Bay Area. The statistics of the three data sets are summarized in TABLE 1.2.

Table 1.2: Statistics of the data sets

Traces	*MIT Reality*	*DieselNet*	*Cabspotting*
Network type	Bluetooth	802.11b	none
No. devices	97	34	536
No. contacts	54,667	2,284	111,153
Duration (days)	246	20	30

1.4.4.2 Experiment Setup

We launch the experiment on the HaggleSim simulator [13]. It takes the discrete sequential encounter events and the corresponding social graph as the inputs and makes data forwarding decisions using various routing algorithms. For each experiment, we emulate 1000 messages with one-week lifetime sent from a random selected source to destination. We run every experiment 20 times for statistical convergence. The following performance metrics are used to evaluate the performance of the routing algorithms.

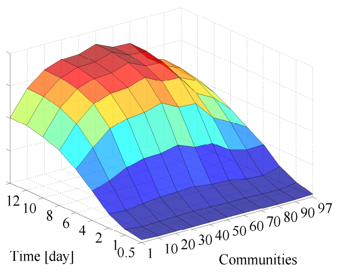
- **Delivery ratio:** the ratio of the number of destinations having received the data to the total number of destinations.
- **Average delay:** the average time delay for each data item delivered from the source to the destination.
- **Average cost:** the average number of relays used for data delivery from the source to the destination.

We extract a 2-week session from MIT Reality, DieselNet and Cabspotting respectively and run the simulator over the selected sessions with uniformly generated traffic. The SMART algorithm is implemented and is compared to other existing routing algorithms.

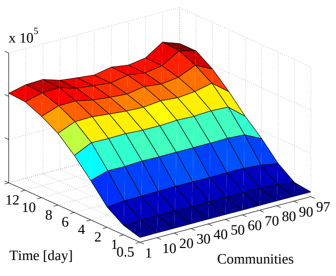
1.4.4.3 Impact of community numbers

We first investigate the impact of the number of communities on the performance of SMART. We apply the proposed m-partition algorithm for community partitioning on the three mobility traces and then use SMART to route messages.

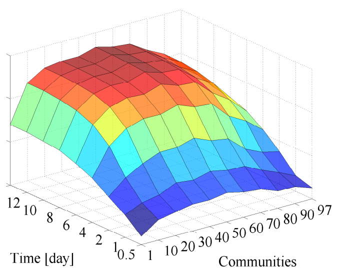
Fig. 1.3 shows the performance metrics as a function of community number m (varying from 1 to the size of the data sets) and time on MIT Reality trace. The delivery ratio of MIT Reality trace is shown in Fig. 1.3(a). According to this figure, when no community partitioning algorithm is applied ($m = 1$), the delivery ratio is quite low and increases slowly with time. As the community number is set to an appropriate value (e.g. $m = 10$), the delivery ratio increases dramatically, which is almost 2 times as much as that when $m = 1$. For $10 \leq m \leq 90$, the delivery ratio becomes stable and has only small fluctuation. When the community number approaches to the size of the data set ($m = 97$), the performance drops dramatically since the impact of the community structure disappears. The average delay is illustrated in Fig. 1.3(b). It



(a) Delivery ratio



(b) Average delay



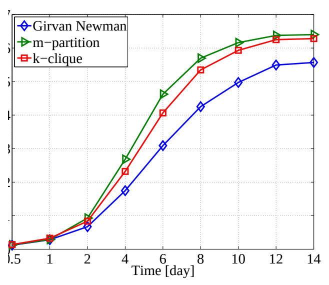
(c) Average cost

Figure 1.3: The performance metrics as a function of community number and time (MIT Reality).

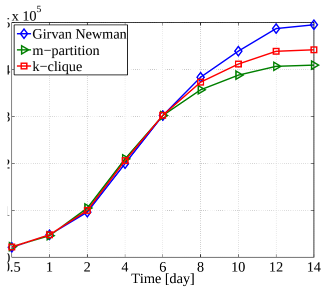
is seen that the average delay is almost the same for all community numbers and it only varies with time. The average cost is shown in 1.3(c). Similar to delivery ratio, the average cost is influenced by m and increases to a stable value when $10 \leq m \leq 90$. Similar results are also found in DieselNet and Cabspotting. The results suggest that SMART performs better when the community structure is outlined, while the performance of SMART is low when no community structure is indicated in the network. It also reveals that the proper value of m is within a wide range. In the rest of our experiments, we fix our community number to $m = 10$.

1.4.4.4 Impact of community partitioning algorithms

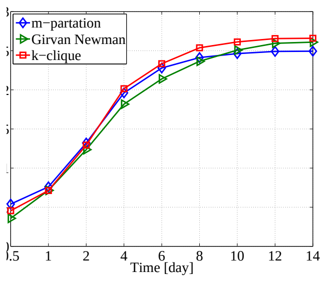
We show the impact of community partitioning algorithms on the SMART routing scheme in this group of experiments. We evaluate the performance of SMART using different community partitioning algorithms, including m-partition, k-clique percolation algorithm [18] (which considers the adjacent k-cliques as communities), and



(a) Delivery ratio



(b) Average delay



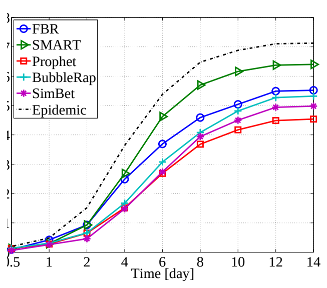
(c) Average cost

Figure 1.4: The performance of SMART under different community partitioning algorithms (MIT Reality).

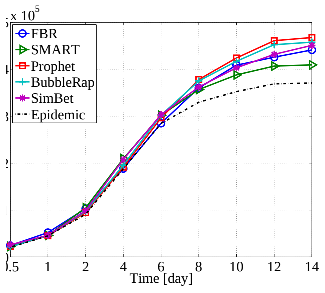
Girvan-Newman algorithm [12] (which continues removing edges with the highest betweenness until a certain threshold is reached).

Fig. 1.4 presents experimental results of the MIT Reality trace. As shown in Fig. 1.4(a), the m-partition method outperforms Girvan Newman by 10% and k-clique percolation by 2% in delivery ratio. In terms of average delay, as shown in Fig. 1.4(b), m-partition performs slightly better than the other two algorithms. The three algorithms produce similar average costs as shown in Fig. 1.4(c). Similar results are also observed in DieselNet and cabspotting data traces.

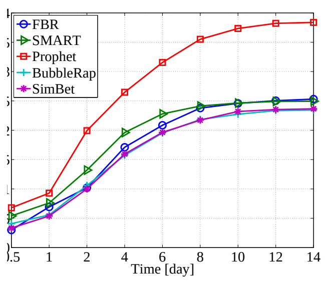
Despite the different algorithms used for community partitioning, the routing performance is quite similar for all three datasets. It indicates that the proposed SMART routing mechanism is not sensitive to community partition. Since Girvan Newman and k-clique percolation need global network topology information which is difficult to obtain in opportunistic networks, the proposed m-partition algorithm is more suitable for distributed implementation in the real world.



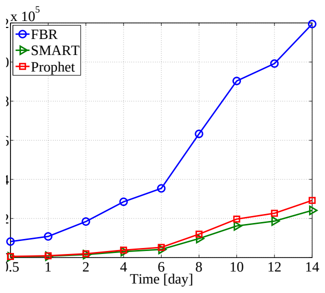
(a) Delivery ratio



(b) Average delay



(c) Average cost



(d) Control overhead

Figure 1.5: The performance comparison of various strategies on MIT Reality Mining trace

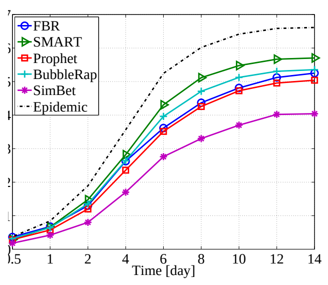
1.4.4.5 Performance comparison

We compare SMART with five existing routing strategies for opportunistic networks: PROPHET [15], SimBet [4], Bubble Rap [14], Friendship Based Routing (FBR) [2], and Epidemic routing [23]. PROPHET is a utility-based strategy according to encounter history. It forwards data to the nodes with higher delivery rate based on contact history. SimBet is a utility-based strategy according to social features. It considers social properties including similarity and centrality to make data forwarding decisions. Bubble Rap is a community-based strategy. It depends on community structure and routes data based on rankings calculated from social centrality. FBR algorithm is another community-based algorithm. It constructs temporal community and uses the nodes with direct connection to the destination community for data delivery. Epidemic routing is a flooding strategy. It has high delivery cost, but its delivery ratio and delay approach the theoretical bounds. To show the amount of traffic in the routing algorithms, we add control overhead as an additional metric to evaluate different routing strategies.

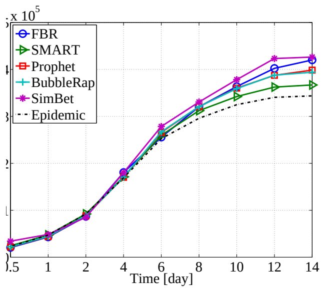
Fig. 1.5 shows the performance of various algorithms as a function of time on MIT Reality trace. The delivery ratio is compared in Fig. 1.5(a). It shows that S-MART outperforms PROPHET, SimBet, FBR and Bubble Rap. The delivery ratio of SMART is about 10% higher compared to those of Bubble Rap and FBR, 15% higher than that of SimBet and nearly 20% higher than that of PROPHET. The reason that PROPHET performs the worst is due to the reason that it fails to adapt to the community structure of the mobility trace. SimBet exploits social properties to enhance the delivery ratio but it can not adapt to the decaying effect of social features. Bubble Rap and FBR take advantages of the community structure, so they perform better than PROPHET, but not as well as SMART. Since Epidemic routing represents the theoretical upper bound of the delivery ratio, the performance of SMART is below that of the Epidemic routing. Average delay is compared in Fig. 1.5(b). Again, the delay of SMART is lower than the other four strategies (most of the time their performance is very close), but higher than the lower bound (Epidemic routing). The average cost is compared in Fig. 1.5(c). The cost of PROPHET is the highest. The cost of SMART is slightly higher than those of the others due to the decaying effect, which makes SMART take more relays for data delivery. The comparison of the control overhead on MIT Reality data trace is shown in Fig. 1.5(d). The overhead of SMART is around 20 KB at the end of the experimental period, which is mainly caused by the exchange of the friend list. PROPHET has over 10% higher overhead than SMART since it needs to exchange transitivity information. FBR takes 4 times more control overhead than SMART because it requires the encounter information from the neighbors.

Fig. 1.6 presents the performance results of various algorithms as a function of time on DieselNet dataset. The delivery ratio is depicted in Fig. 1.6(a). SMART outperforms Bubble Rap by 3%, FBR by 5% and PROPHET by 8%. It has nearly 20% higher delivery ratio than SimBet. Regarding the average delay and the average cost of each strategy shown in Fig. 1.6(b) and Fig. 1.6(c), SMART has very close average delay to Epidemic, which is less than those in the other strategies. The average cost of SMART is about 50% of that of PROPHET and higher than those of FBR and SimBet. DieselNet has very similar network structure with MIT Reality and thus has similar trend on delivery ratio with MIT Reality. However, due to the regular and repetition routine of buses in DieselNet, it makes the SimBet meet dead ends quite often and takes more time to wait until reaching the destinations. Therefore, it has lower delivery ratio and higher average cost. Since DieselNet has tighter clustering structure, it makes Bubble Rap and FBR perform close to SMART. SMART has similar cost with social-related strategies but much lower cost than PROPHET. The overhead is presented in Fig. 1.6(d). The overhead of SMART is the lowest, which is 4KB at the end of the experimental period. PROPHET has 25% higher overhead than SMART and the overhead of FBR is 2.5 times of that of SMART.

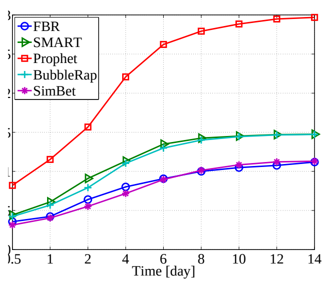
The comparison of the different different algorithms' performance on Cabspotting trace is shown in Fig. 1.7. Fig. 1.7(a) depicts the delivery ratio of the various algorithms as a function of time. The SMART algorithm has very similar performance as PROPHET. It outperforms FBR by 5%. The Bubble Rap algorithm is affected by weak community structure, which lowers down its delivery ratio around 10% com-



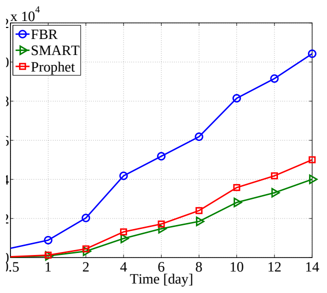
(a) Delivery ratio



(b) Average delay



(c) Average cost



(d) Control overhead

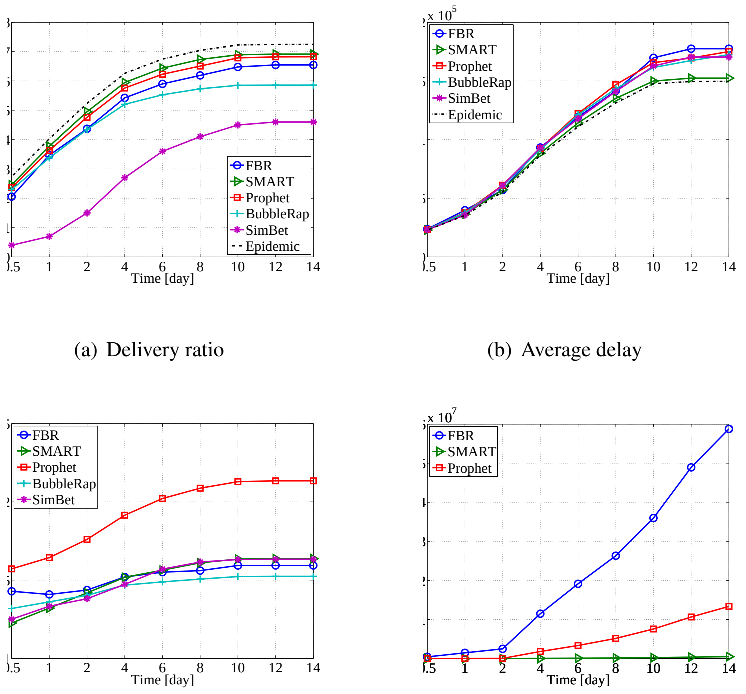
Figure 1.6: The performance comparison of various strategies on DieselNet trace

pared to SMART. SimBet has the lowest delivery ratio, which is much lower than other strategies. In terms of the average delay shown in Fig. 1.7(b), the delay of S-MART is as low as that of Epidemic and is much lower than those of others. The average costs of various algorithms are similar as shown in Fig. 1.7(c). The overhead is shown in Fig. 1.7(d). The overhead of SMART is around 20 KB at the end of the experimental period, while the overhead of FBR and PROPHET is much higher than that of SMART.

In summary, the proposed SMART strategy outperforms the utility-based and community-based strategies on various opportunistic network datasets in most of the performance metrics.

1.5 Multicast

Multicast refers to one-to-many communication in a network. The sender may specify a group of destination addresses for a single piece of information. For multicast



(a) Delivery ratio (b) Average delay

(c) Average cost (d) Control overhead

Figure 1.7: The performance comparison of various strategies on Cabspotting trace

in opportunistic networks, it also applies the "store-carry-forward" approach to relay information among mobile nodes. Different from unicast, when a node encounters another node, it decides whether to create a new copy of the data, or simply hands over the data to the other nodes. In multicast, more and more copies of the original information are created and propagated in the network, until all the destinations receive a copy of the original information.

In multicast, a message holder is expected to forward a message to multiple destinations. To reduce the overhead and forwarding cost, the destinations will share the routing path until the point that they have to be separated, which usually results in a tree structure. The basic idea of our scheme is to improve multicast efficiency by exploring node social features and community structure. We first introduce the definition of *dynamic social features*, which represents the dynamic social characteristics of mobile users regarding their social profiles and encountering behaviors. And then we present *enhanced dynamic social features*, an improved version of dynamic social features. Based on the enhanced dynamic social features, we apply the compare-split scheme using community detection to form a multicast tree. Specifically, we propose two ***C***ommunity and ***S***ocial feature-based multicast algorithms named *Multi-CSDO* that involves ***D***estination nodes ***O***nly in community detection and *Multi-CSDR* that involves both the ***D***estination nodes and the ***R***elay candidates in community detection in case the relay candidates are also socially similar. The details are presented as follows.

1.5.1 Dynamic social features

Suppose the social profile of a mobile user is represented by m social features $\langle E_1, E_2, \cdots, E_m \rangle$. The social profile similarity can be calculated using equation (1.1). However, such social features are static and cannot represent the encountering behavior of mobile users. In this section, we introduce the dynamic social features and enhanced dynamic social features to incorporate the behavior dynamics.

Definition 1.6 Dynamic social features Assume a mobile node x has social profiles $E_x(\tau) = < E_{x1}(\tau), E_{x2}(\tau), \cdots, E_{xm}(\tau) >$, its dynamic social features can be represented by a vector $\tilde{E}_x(\tau) = < \tilde{E}_{x1}(\tau), \tilde{E}_{x2}(\tau), \cdots, \tilde{E}_{xm}(\tau) >$, where $\tilde{E}_{xk}$ $(0 \leq \tilde{E}_{xk} \leq 1)$ is the ratio of node x encountering other nodes having the same social profile E_{xk}, which is computed by

$$\tilde{E}_{xk}(\tau) = \frac{M_k(\tau)}{M_{total}(\tau)}. \tag{1.11}$$

Here, $M_k(\tau)$ is the number of meetings of node x with other nodes having social feature value E_{xk}, and $M_{total}(\tau)$ is the total number of nodes x has met in the duration τ.

Dynamic social features not only record if a node has certain social feature values, but also record the frequency this node has met other nodes with the same social feature values. Unlike the static ones, they are time-related and adjusted to the user contact behavior change over time.

The definition of dynamic social features is based on frequency, which cannot distinguish the cases, for example, if A has met 1 *Student* out of 2 people it has met in total and B has met 5 *Students* out of 10 people it has met in total in the history we observe. Both of them have the same frequency of $1/2$ to meet a *Student*, but B is more active in meeting people. To break the tie and favor the more active node, there are many formulas we can design. Here, we come up with the following enhanced dynamic social features to serve our purposes.

Definition 1.7 Enhanced dynamic social features Following the notations in definition 1.6, the enhanced dynamic social features of mobile node x are defined as $\hat{E}_x(\tau) = < \hat{E}_{x1}(\tau), \hat{E}_{x2}(\tau), \cdots, \hat{E}_{xm}(\tau) >$, where

$$\hat{E}_{xk}(\tau) = (\frac{M_k(\tau)+1}{M_{total}(\tau)+1})^{p_k} * (\frac{M_k(\tau)}{M_{total}(\tau)+1})^{1-p_k}, \tag{1.12}$$

where $p_k = \frac{M_k(\tau)}{M_{total}(\tau)}$.

This definition predicts $\hat{E}_{xk}$ by looking at the next meeting probability of node x with another node having the same social feature value. In the next time, the total meeting times will be $M_{total}(\tau)+1$. The first part $(\frac{M_k(\tau)+1}{M_{total}(\tau)+1})^{p_k}$ means that there will be p_k probability that x will have a "good" meeting with another node having the same social feature value next time. In this case, M_k will also be incremented by 1. The second part $(\frac{M_k(\tau)}{M_{total}(\tau)+1})^{1-p_k}$ means that there will be $1-p_k$ probability for x not to meet a node with the same social feature value next time. In that case, M_k will remain the same. The definition for $\hat{E}_{xk}$ then takes the geometric mean of the two parts.

With the above definitions, the social similarity of two nodes x and y regarding their enhanced dynamic social features is defined as follows.

Definition 1.8 Enhanced social similarity Following the notations in definition 1.7, the enhanced social similarity of node x and y is computed by their Euclidean distance subtracting from 1:

$$S_{x,y}(\tau) = 1 - \frac{\sqrt{\sum_{k=1}^{m}(\hat{E}_{yk}(\tau) - \hat{E}_{xk}(\tau))^2}}{\sqrt{m}}. \tag{1.13}$$

Based on the enhanced dynamic social features and similarity measurement, we proposed two multicast algorithms in the following subsections.

1.5.2 *The* Multi-CSDO *algorithm*

The first proposed multicast algorithm is called Multi-CSDO as shown in Fig. 1.8. Its basic idea is as follows: First, a source node s has a destination set to multicast a message to and s is the initial message holder or relay node x. When x meets a node y, if y is one of the destinations, y gets the message and is removed from the destination set. Next we use a compare-split scheme to make a decision of whether it is better to pass some destinations to y. Both x and y are called relay candidates in the decision. To separate the destinations into x's community or y's community, we use a community detection algorithm involving only the destination nodes based on their social similarities. The community detection algorithm we use takes a distance matrix coming from a similarity weighted graph as an input. The following are the details.

1.5.2.1 *Similarity weighted graph and distance matrix*

In Multi-CSDO, as shown in an example in Fig. 1.9, when a message holder x encounters a node y, we construct a similarity weighted graph involving only the destination nodes. The weight of the edges is the social similarity of the two connected destination nodes calculated using static social features (denoted by dashed edges in

Algorithm Multi-CSDO: community and social feature-based multicast involving destinations only in community detection

```
Require: The source node $s$ and its destination set $D_s = \{d_1, d_2, \cdots, d_n\}$; $s$ is the
    initial message holder $x$
 1: while not all of the destinations receive the message do
 2:     On contact between a message holder $x$ and node $y$:
 3:     if $y \in D_x$ then
 4:         /* Found destination $y$ */
 5:         $y$ gets the message and $x$ removes $y$ from $D_x$
 6:     end if
 7:     /* Compare node social similarity and split the destinations */
 8:     Construct a weighted graph and a distance matrix of the destination nodes
        only as explained in Section 1.5.2
 9:     Feed the distance matrix to the hierarchical clustering algorithm to generate
        two communities $C_1$ and $C_2$ as explained in Section 1.5.2
10:     Compare the social similarity of $C_1$ and $C_2$ with $x$ and $y$ using enhanced dy-
        namic social features, respectively
11:     Whichever ($x$ or $y$) is more socially similar to each of the communities will be
        the message carrier for that community
12: end while
```

Figure 1.8: The Multi-CSDO multicast algorithm.

Fig. 1.9) as their dynamic social features are not known to the relay candidates in a distributed algorithm. With the similarity weighted graph, we can create a distance matrix as shown in Fig. 1.10 to indicate the social difference or distance between each pair of destinations. The social distance between two destinations d_i and d_j is defined as $1 - S(d_i, d_j)$ here. The distance matrix will be used in the following community detection algorithm to separate the destinations into two communities.

1.5.2.2 Community detection algorithm

We use a hierarchical clustering algorithm called complete-linkage clustering [5] to split the destinations into two communities. We choose this one because it best matches our needs and there is an existing Python package [1] available for this algorithm so that we do not have to reinvent the wheel.

The idea of the complete-linkage hierarchical community detection algorithm we adopt is as follows: At the beginning of the process, each node is in a community of its own. The communities are then sequentially combined into larger communities, until all nodes end up being in one community. At each step, the two communities separated by the shortest distance are combined. The distance between communities is defined as the distance between those two nodes (one in each community) that are farthest away from each other. We feed our distance matrix and the number of communities 2 into the package and obtain two communities as the result.

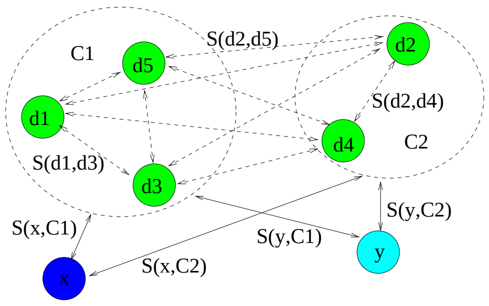


Figure 1.9: The similarity weighted graph and community detection involving destination nodes only. Node x is a message holder and y is a newly met node. The green nodes are the destinations. The weight of a dashed edge is the social similarity calculated using static social features while the weight of a solid edge is the social similarity calculated using the enhanced dynamic social features. The destinations are split into two communities C_1 and C_2 based on their social similarities.

	d1	d2	d3	d4	•••
d1	0	1−S(d1,d2)	1−S(d1,d3)	1−S(d1,d4)	
d2	1−S(d1,d2)	0	1−S(d2,d3)	1−S(d2,d4)	
d3	1−S(d1,d3)	1−S(d2,d3)	0	1−S(d3,d4)	
d4	1−S(d1,d4)	1−S(d2,d4)	1−S(d3,d4)	0	
⋮					

Figure 1.10: The distance matrix. The distance between destination nodes d_i and d_j is $1 - S(d_i, d_j)$ if $i \neq j$; otherwise 0.

1.5.2.3 Destinations split

After applying the community detection algorithm, the destinations are separated into two communities C_1 and C_2. Next we decide which relay candidate, x or y, should carry the destinations in which community. We compare the social similarity of each relay candidate with each community using enhanced dynamic social features (denoted by the solid edges in Fig. 1.9). The social similarity between a node and a community should include all of the social feature values of the nodes involved. After calculation, whichever is more socially similar to a community will be the relay node for the destinations in that community.

In Multi-CSDO, x and y are supposed to be in different communities, which may not be true if they are socially similar. Thus, in the next section, we introduce the Multi-CSDR algorithm by incorporating both x and y in the community detection and make our decision more accurate by considering more node relationships.

1.5.3 *The* Multi-CSDR *algorithm*

The second multicast algorithm is called Multi-CSDR. It has a similar structure with the first algorithm, but has several differences. As shown in the example in Fig. 1.11, first, the community detection algorithm involves both the destination nodes and the relay candidates x and y. Thus the similarity weighted graph adds the social similarity between each relay candidate and each destination node. The social similarity between two destination nodes is still calculated using static social features and is denoted by a dashed edge in Fig. 1.11. However, the social similarity between a relay candidate and a destination is calculated using enhanced dynamic social features as they can be obtained and is denoted by a solid edge in Fig. 1.11. We still use the same community detection algorithm. But the distance matrix now also includes the distance between each relay candidate and each destination. After applying the community detection algorithm, the destinations in x's community will be carried by x and those in y's will be carried by y. For other cases, for example, if x and y are in the same community, then x will still be the carrier for the original destination set.

In this algorithm, by adding the social similarity of each relay node with each destination using enhanced dynamic social features, we hope to improve the accuracy of the compare-split scheme.

1.5.4 Performance Evaluation

We evaluate the performance of the proposed multicast algorithms by comparing them with the existing ones using a simulator written in Python. The simulations were conducted using a real conference trace [22] reflecting an opportunistic network created at IEEE Infocom 2006 in Miami. The trace recorded conference attenders' encounter history using Bluetooth small devices (iMotes) for four days at the conference. The trace dataset consists of two parts: *contacts* between iMote devices that were carried by participants and self-reported *social features* of the participants collected using a questionnaire form. The six social features extracted from the dataset

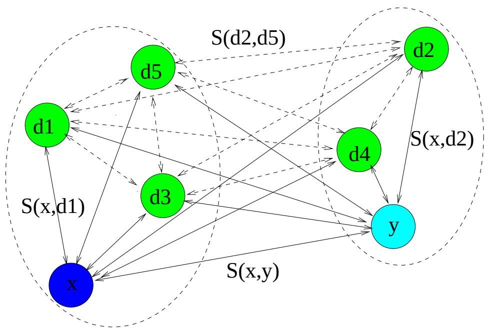


Figure 1.11: The similarity weighted graph and community detection involving both destination nodes and relay candidates *x* and *y*. Node *x* is a message holder and *y* is a newly met node. The green nodes are the destinations. The weight of a dashed edge is the social similarity calculated using static social features while the weight of a solid edge is the social similarity calculated using the enhanced dynamic social features. The nodes are split into two communities based on their social similarities.

were *Affiliation, City, Nationality, Language, Country,* and *Position*. In this trace, 62 nodes with complete social feature information were considered in the simulation.

We compare our algorithm with the following existing multicast algorithms.

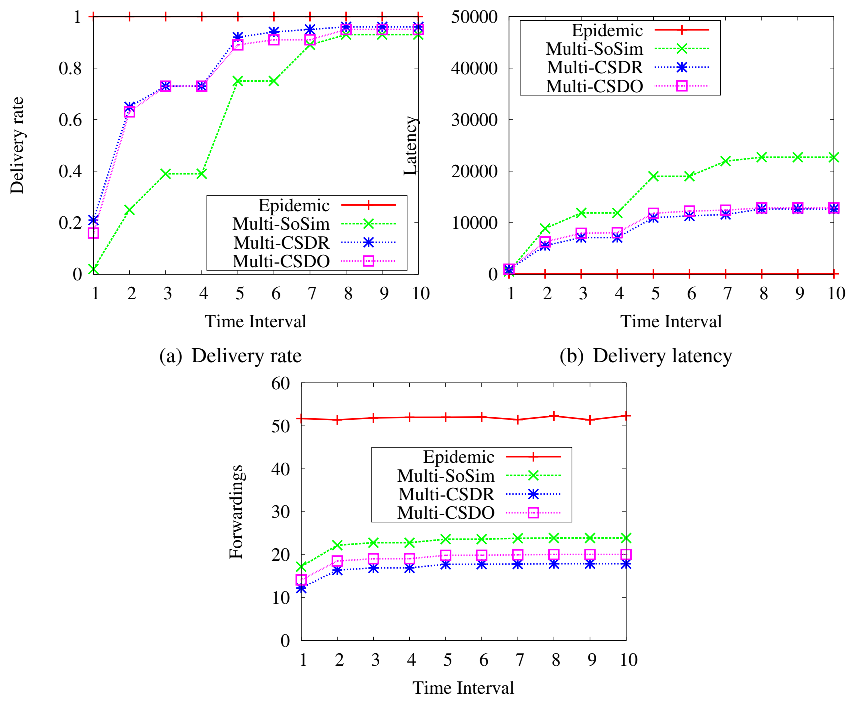
- **The Epidemic Algorithm (Epidemic) [23]** : The message is spread epidemically throughout the network until it reaches all of the destinations.
- **The Social-Similarity-based Multicast Algorithm (Multi-Sosim) [27]**: The multicast algorithm based on dynamic social features in our previous work.

Three important metrics are used to evaluate the performance of the multicast algorithms.

- **Delivery rate:** The ratio of the number of successful multicasts (where the message is delivered to all the destinations) to the number of total multicasts generated.
- **Delivery latency:** The time from the start of multicast to when all of the multicast destinations have received the message.
- **Number of forwardings:** The number of hops needed to deliver a message to all of the multicast destinations.

1.5.5 Simulation setup

In our simulations, we divided the whole trace time into 10 intervals. Thus, 1 time interval is $1/10$ of the total time length. For each algorithm, we tried 5 and 10 des-



(a) Delivery rate (b) Delivery latency

(c) Number of forwardings

Figure 1.12: Comparison of different algorithms with 5 destinations using all devices in the trace

tinations. In each experiment, we randomly generated a source and its destination set. Since the whole trace only contains four days of node contact history, the time interval we observed to calculate the dynamic and enhanced dynamic social features was counted from the beginning of the trace up until the time we needed to make a routing decision. For the community detection algorithm, we adopted the Python package available at [1] for the complete-linkage hierarchical clustering algorithm. We ran each algorithm 300 times and averaged the results.

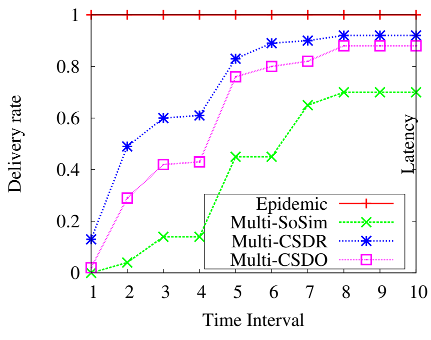
1.5.6 Simulation results

The simulation results comparing our algorithms with others using 5 and 10 destinations are shown in Figs. 1.12 and 1.13, respectively. For the Epidemic algorithm, as expected, it has the highest delivery rate (100%) and lowest delivery latency (almost close to 0) but highest number of forwardings.

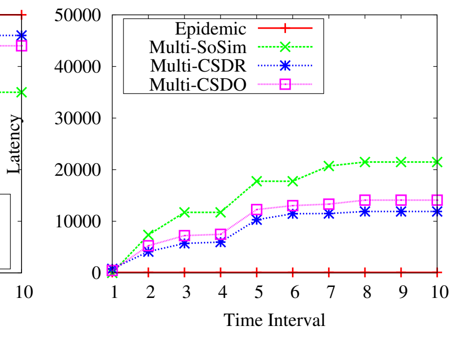
With both 5 and 10 destinations, Multi-CSDO and Multi-CSDR consistently outperform Multi-Sosim in terms of delivery rate, latency, and number of forwardings. In the 5 destination case, Multi-CSDR and Multi-CSDO improve the delivery rate of Multi-Sosim by as much as 9 times and 7 times, respectively, reduce the latency of Multi-Sosim by as much as 47% and 43%, respectively, and decrease the number of forwardings of Multi-Sosim by as much as 29% and 18%, respectively. Similarly, in the 10 destination case, Multi-CSDR and Multi-CSDO improve the delivery rate of Multi-Sosim by as much as 11 times and 6 times, respectively, reduce the latency of Multi-Sosim by as much as 51% and 38%, respectively, and decrease the number of forwardings of Multi-Sosim by as much as 40% and 28%, respectively. These tell us that adding the social relationships among destinations in the compare-split scheme can facilitate multicast.

Furthermore, Multi-CSDR has better delivery rate, lower latency, and lower number of forwardings than Multi-CSDO with both 5 and 10 destinations. In the 5 destination case, Multi-CSDR improves the delivery rate of Multi-CSDO by as much as 5%, reduce the latency of Multi-CSDO by as much as 11%, and decrease the number of forwardings of Multi-CSDO by as much as 13%. Similarly, in the 10 destination case, Multi-CSDR improves the delivery rate of Multi-CSDO by as much as 69%, reduce the latency of Multi-CSDO by as much as 22%, and decrease the number of forwardings of Multi-CSDO by as much as 17%. These verify that considering the social relationship between each relay candidate and each destination, and calculating their social similarity using enhanced dynamic social features can improve multicast performance.

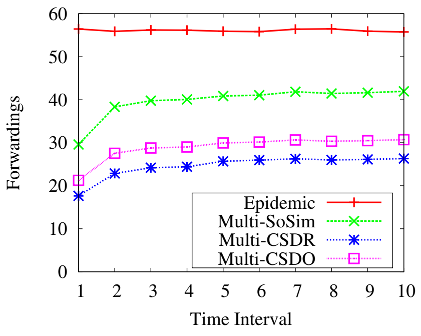
In summary, these results confirm that obtaining more accurate dynamic information and using better compare-split schemes can make multicast more efficient.



(a) Delivery rate



(b) Delivery latency



(c) Number of forwardings

Figure 1.13: Comparison of different algorithms with 10 **destinations using all devices in the trace**

1.6 Conclusion

In this chapter, we studied information dissemination mechanisms in opportunistic networks. Based on the coupling of opportunistic networks and human social networks, we modeled the opportunistic network as a social graph, and exploited social features such as users' social profiles, social relationships and network structures to build more efficient information dissemination schemes.

Specifically, we explored the unicast and multicast information dissemination in opportunistic networks. For one-to-one communication (unicast), we proposed a social- and mobile-aware message routing strategy called SMART. In this strategy, an opportunistic network is divided into a number of communities using adaptive community partitioning algorithms. Two data routing processes were introduced: intra-community communications and inter-community communications. It was shown that SMART adopts both community structure and social similarities to enhance data forwarding efficiency. Extensive trace-driven experiments showed that SMART outperforms other unicast strategies in various opportunistic network traces. For one-to-many communication, we proposed two social feature-based multicast algorithms named Multi-CSDO and Multi-CSDR. The proposed algorithms used enhanced dynamic social features to capture nodes' contact behavior, and applied a compare-split scheme based on community detection to select the best relay node for multiple destinations in each hop to improve multicast efficiency. Simulations using a real trace of a mobile social network showed that the proposed algorithms outperform the existing ones in various performance metrics.

Our work revealed the possibility of exploiting social features to facilitate communication networks. By incorporating the social properties such as centrality, similarity, and community structure into algorithm and protocol design, the efficiency of information dissemination in mobile and wireless network achieved great improvement.

1.7 Glossary

Opportunistic network: A sparse dynamic wireless network where mobile nodes work on ad hoc mode and can communicate with each other when they move into their communication range.

Mobile Social Network: A mobile communication system focusing not only on the interactions but also on the social aspects of the users.

Unicast: One-to-one communication in opportunistic networks, where one mobile node sends information to another node.

Multicast: One-to-many communication where where one mobile node sends information to a set of destinations.

References

[1] Hierarchical clustering. http://docs.scipy.org/doc/scipy/reference/cluster.hierarchy.html.

[2] E. Bulut and B. Szymanski. Exploiting friendship relations for efficient routing in mobile social networks. *IEEE Transactions on Parallel and Distributed Systems*, PP(99):1, 2012.

[3] John Burgess, Brian Neil Levine, Ratul Mahajan, John Zahorjan, Aruna Balasubramanian, Arun Venkataramani, Yun Zhou, Bruce Croft, Nilanjan Banerjee, Mark Corner, and Don Towsley. CRAWDAD data set umass/diesel (v. 2008-09-14). http://crawdad.cs.dartmouth.edu/umass/diesel, 2008.

[4] Elizabeth M. Daly and Mads Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc '07)*, pages 32–40, New York, NY, USA, 2007. ACM.

[5] D. Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 1977.

[6] C.H.Q. Ding, Xiaofeng He, Hongyuan Zha, Ming Gu, and H.D. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *IEEE ICDM'01*, pages 107–114, 2001.

[7] Nathan Eagle and Alex (Sandy) Pentland. CRAWDAD data set mit/reality (v. 2005-07-01). http://crawdad.cs.dartmouth.edu/mit/reality, 2005.

[8] Kevin Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '03)*, pages 27–34, New York, NY, USA, 2003. ACM.

[9] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3C5):75 – 174, 2010.

[10] L. Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40(1):35–41, 1977.

[11] L. Freeman. Centrality in social networks: Conceptual clarification. *Social Networks*, 1(3):215–239, 1979.

[12] M. Girvan and M. E. J. Newman. A Set of Measures of Centrality Based on Betweenness. *PNAS*, 99(12):7821–7826, 2002.

[13] Pan Hui and Jon Crowcroft. How small labels create big improvements. In *PerCom Workshops'07*, pages 65 –70, 2007.

[14] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc '08)*, pages 241–250, New York, NY, USA, 2008. ACM.

[15] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic Routing in Intermittently Connected Networks. volume 3126 of *Lecture Notes in Computer Science*, pages 239–254. Springer-Verlag GmbH, 2004.

[16] Mark EJ Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[17] Nam P. Nguyen, Thang N. Dinh, Ying Xuan, and My T. Thai. Adaptive algorithms for detecting community structure in dynamic social networks. In *IEEE INFOCOM '11*, pages 2282–2290, 2011.

[18] Gergely Palla, Imre Derenyi, Illes Farkas, and Tamas Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814, 2005.

[19] Michal Piorkowski, Natasa Sarafijanovic-Djukic, and Matthias Grossglauser. CRAWDAD data set epfl/mobility (v. 2009-02-24). http://crawdad.cs.dartmouth.edu/epfl/mobility, 2009.

[20] U. N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 79(3):7821–7826, 2007.

[21] J. Reichardt. *Structure in Complex Networks*. Lecture Notes in Physics, 2009.

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

[23] Amin Vahdat, David Becker, et al. Epidemic routing for partially connected ad hoc networks. Technical report, CS-200006, Duke University, 2000.

[24] N. Vastardis and Kun Yang. Mobile social networks: Architectures, social properties, and key research challenges. *IEEE Communications Surveys Tutorials*, 15(3):1355–1371, Third 2013.

[25] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[26] Jie Wu and Yunsheng Wang. Social feature-based multi-path routing in delay tolerant networks. In *Proceedings of IEEE INFOCOM'12*, pages 1368–1376, March 2012.

[27] Y. Xu and X Chen. Social-similarity-based multicast algorithm in impromptu mobile social networks. In *IEEE Globecom*, 2014.

[28] Konglin Zhu, Wenzhong Li, and Xiaoming Fu. SMART: A Social- and Mobile-Aware Routing Strategy for Disruption-Tolerant Networks. *IEEE Transactions on Vehicular Technology*, 63(7):3423–3434, Sept 2014.