On Social Delay Tolerant Networking: Aggregation, Tie Detection, and Routing

Kaimin Wei, Deze Zeng, *Member, IEEE,* Song Guo , *Senior Member, IEEE,* and Ke Xu

Abstract—Social-based routing protocols have shown their promising capability to improve the message delivery efficiency in Delay Tolerant Networks (DTNs). The efficiency greatly relies on the quality of the aggregated social graph that is determined by the metrics used to measure the strength of social connections. In this paper, we propose an improved metrics that leads to high-quality social graph by taking both frequency and duration of contacts into consideration. Furthermore, to improve the performance of social-based message transmission, we systematically study the community evolution problem that has been little investigated in the literation. Distributed algorithms based on our new proposed metrics are developed such that the overlapping communities and bridge nodes (i.e., connecting nodes between communities) can be dynamically detected in an evolutionary social network. Finally, we take all the results above into our social-based routing design. Extensive trace-driven simulation results show that our routing algorithm outperforms existing social-based forwarding strategies significantly.

Index Terms—Social-aware routing, connection strength metric, social graph, community, bridge node.

1 Introduction

DELAY-Tolerant Networks (DTNs) [1] emerge as an indispensable complement to the traditional wireless network to support a variety of delay-tolerant applications. They are characterized by the lack of contemporaneous and continuous end-to-end connections, but only opportunistically intermittent connections. In such networks, opportunistic routing [2]–[9] has attracted much attention because its "carry-and-forward" data dissemination scheme well exploits the unexpected transmission opportunities. As its simplest form, epidemic routing [3] allows messages to be aggressively replicated and forwarded to each encountered node, incurring high resource (e.g, energy, buffer, etc.) consumption. To address this issue, various improved opportunistic routing schemes [4], [5] have been proposed by exploring a critical issue on how to select appropriate intermediate nodes, i.e., relay nodes, such that messages are forwarded only to them, other than blindly to all encountered ones.

It has been discovered that some DTNs like mobile social network (MSN) [10] exhibit human behaviors, whose benefits for relay node selection have been validated by several social-based routing protocols such as BubbleRap [4], PeopleRank [5] and SimBet [11]. To exploit the social network property in these DTNs, we take a systematical approach by developing the following techniques that are of great importance: (1) high-quality social graph aggregation, (2) dynamic detection of social ties, and (3) efficient social-aware routing, in which, furthermore, one serves as an underlying basis for the next.

Existing routing protocols for social DTNs overlook the construction of high-quality social graph [12]. In particular, without accurate measurement of the connections between nodes, some of them even simplify treat simple social graph as contact graph. The limitation of existing metrics for connection strength motivates us to re-examine how to define an improved one for high-quality social graph aggregation. We then invent a hybrid connection strength metrics and incorporate it into a distributed density-based aggregation approach to build social graph. Because of his strong descriptive capability in distinguishing connection strength, even existing social-based routing protocols (e.g., BubbleRap [4] and PeopleRank [5]) can benefit by adopting our proposed metrics with a better performance than their original versions.

The quality of social ties also plays an essential role to the performance of social-aware routing protocols. Existing community detection methods, however, can hardly uncover an accurate community structure of a DTN since they either rely on empirical threshold values [13] or ignore community evolutions (e.g., nodes leaving from a community) [14]. Even worse, how to identify and exploit the bridge nodes for social-based routing, especially for inter-community communications, has been little studied in the literature. In contrast, we develop a distributed community detection approach to uncover overlapping communities and capture their evolutions, without any predetermined empirical value. Meanwhile, a self-recommending approach is proposed for bridge node identification.

The final step is to exploit the extracted social ties for social-aware DTN routing. In particular, forwarding candidate set (i.e., the messages that shall be forwarded)

- *K. Wei and K. Xu are with the State Key Lab of Software Development Environment, Beihang University, China.*
  E-mail: kexu@nlsde.buaa.edu.cn
- *D. Zeng and S. Guo are with the Department of Computer Science and Engineering, The University of Aizu, Japan.*
  E-mail: sguo@u-aizu.ac.jp

and their forwarding order should be carefully selected such that the message delivery probability is maximized. However, this issue is always overlooked in existing social-based routing protocols [4], [5], [15], in which the buffered messages are randomly chosen and then greedily forwarded node one by one until the duration ends. To deal with this issue, we we propose a connection-strength aware routing (CSAR) protocol, in which the message utility is defined to determine the forwarding set and order. We further explore the benefit of the detected community structure and apply the *inter-community multi-copy* and *intra-community single-copy* policy in the consideration of both delivery performance and resource consumption. Extensive real-trace driven simulations show that CSAR achieves the delivery performance closing to epidemic routing and significantly outperforms existing social-based routing protocols. For example, CSAR can achieve a higher message delivery ratio by 50% and 65%, but a lower overhead by 18% and 29% than BubbleRap and PeopleRank, respectively.

The remainder of this paper is structured as follows. Section 2 presents related work. Section 3 introduces the system model. Section 4 proposes a new connection metric for social graph aggregation. Section 5 elaborates our distributed algorithms for community and bridge detection. Section 6 presents our CSAR protocol. Section 7 evaluates and analyzes the simulation results. Finally, Section 8 concludes this paper. The specifications of algorithms proposed in the paper are given in the supplementary file.

2 Related Work

2.1 Social Graph Aggregation

To obtain high-quality social graph, several social graph aggregation algorithms have been developed and can be generally categorized into time-based and density-based classes. BubbleRap [4] is a time-based approach which aggregates contacts appeared in a sliding window. However, it is difficult to determine the window length. The density-based approach [5] adopts a connection metric that chooses the appropriate connections to build a social graph. Compared to the time-based class, the density-based one aggregates the desired contacts, other than simply any contacts, into the social graph. While density-based algorithms achieve better performance as revealed in [12], their adopted metrics (e.g., contact frequency), which characterizes the *connection strength* for desired contact selection, has limitation in describing the factor of contact duration that would be critical to the performance as well.

2.2 Social Tie Detection

Social graph describes social ties in terms of, for example, community structure [4], [16] and bridge (i.e., connecting nodes between communities) [11], [17]. On the community detection, Santo et al. [18] give a survey on various community detection approaches. It can be seen that most existing community detection approaches operate in a centralized manner, making them hard to be directly applied to DTNs. To address this issue, distributed community detection algorithms were proposed recently. Hui et al. [4] use k-clique and weighted network approaches to detect overlapping communities. In [13], they further introduce three distributed community detection approaches. All these approaches require a predetermined familiar set threshold and fail to discover community evolutions, and even some of them can not uncover overlapping communities. Later on, Li et al. [14] propose a distributed algorithm to detect overlapping communities without any predetermined threshold. However, it only focuses on community creation and therefore still can not discover any community evolutions.

2.3 Social-Aware Routing

The potential of social ties on the guidance of DTN routing (e.g., relay node selection) has been widely explored by a diversity of social-based routing protocols. Daly et al. [11] propose SimBet which takes advantage of betweenness centrality and similarity to make forwarding decisions. Hui et al. [4] present BubbleRap which forwards messages to the nodes with increasingly large betweenness centrality until they reach destination communities. Later on, PeopleRank [5] makes use of past encounter information to rank nodes and delivers messages to the nodes with a higher utility value. User-Centric [19] takes the social contact patterns and mobile user interests into relay node selection. Furthermore, existing work focuses on how to guide the routing of one message from a single unicast session. When multiple messages to different destinations coexist, due to the limitation and unexpectation of contact duration, it is impossible to guarantee that every message proceeds normally as the guidance in routing. Therefore, it is inevitable to investigate the management of multiple messages. However, this is usually overlooked in existing work.

3 System Model

We consider a category of DTNs exhibiting certain social features like MSN [10], where the communication devices are attached and moved with human beings. The transmission between any two mobile nodes happens only when they move into the reciprocal communication range of each other. Mobile nodes evolve into communities according to their social properties because their mobility is of long-term regularities. Such type of DTN is represented by a social graph $G = (V, E)$, where vertex set V and undirected edge set E denote all mobile nodes and their pairwise social connections, respectively. A social graph is constructed by aggregating all historical contact information. To implement in a distributed manner, each mobile node i maintains a local social graph

$G_i^t = (V_i^t, E_i^t)$ ($V_i^t \subseteq V$ and $E_i^t \subseteq E$) over time t, in which the neighbor set of i is denoted as N_i^t ($N_i^t \subseteq V_i^t$) .

Based on the social graph, social ties like community structure and bridge nodes could be detected. A community is a group of nodes with distinguishable features (e.g., common interest) from others outside. Let $\mathbb{C}_i^t$ be the set of all communities detected by node i at time t. For any community $C \in \mathbb{C}_i^t$, it is denoted as $C = (V_C^t, E_C^t)$, where $V_C^t \subseteq V$ and $E_C^t \subseteq E$.

In general, communities are not isolated but interconnected by edges or common nodes, in which any involving node is called a *bridge node* and can facilitate inter-community communications. Let $B(C, C')$ indicate the set of bridge nodes connecting communities C and C'. Any node i may serve as bridge for multiple community pairs, denoted as $\mathbb{B}_i$.

It is well accepted that bridging centrality [20] is an efficient metric to measure how well a node is located between connected communities. In [21], bridging centrality of a node i is defined as:

$$BC_i = \Phi(i) \cdot \Psi(i), \tag{1}$$

where $\Phi(i)$ and $\Psi(i)$ are betweenness centrality and bridging coefficient, respectively. Betweenness centrality $\Phi(i)$ is defined as

$$\Phi(i) = \sum_{s \neq i \neq t \in V} \frac{\sigma_{st}(i)}{\sigma_{st}}, \tag{2}$$

where σ_{st} is the number of shortest paths between nodes s and t, and $\sigma_{st}(v)$ is the number of shortest paths passing through a node i out of σ_{st}. Bridging coefficient $\Psi(i)$ can be calculated as

$$\Psi(i) = \frac{1}{d(i)} \sum_{v \in N_i} \frac{\delta(v)}{d(v) - 1}, \tag{3}$$

where $d(i)$ is the degree of node i and $\delta(v)$ is the number of edges leaving the neighbors of node v among the edges incident to each direct neighbor v of node i. We notice that $\Psi(i)$ is determined by the connection degree (i.e., number of neighbors) of node i in the social graph.

4 Social Graph Aggregation

In this section, we propose a new connection strength metric that can accurately measure the potential transmission capabilities between nodes in DTNs. Based on this metric, a density-based distributed social graph aggregation approach is applied to construct a social graph on each mobile node.

4.1 Contact Frequency vs. Contact Duration

In density-based aggregation approach, not all but only the selected connections will be included into a social graph such that this social graph shall be maintained at a certain density to prevent the forwarding decision from degenerating to random [12]. As a result, connection strength is used as a metric to determine whether a connection shall be added into the social graph. It can be derived from historical contact information, e.g., *contact frequency* and *average contact time* [12] [14]. The former, denoting the number that two nodes encounter in a period of time, does well in evaluating frequent contacts of mobile nodes but with short duration. The latter, representing the average duration of each encounter for two nodes, is good at assessing occasional contacts with long duration. However, these metrics cannot measure contacts well when the encounter behaviors are complex.

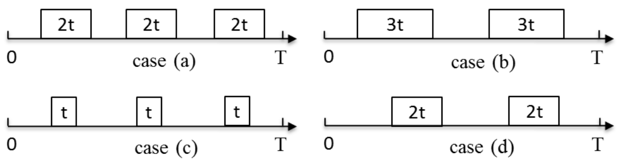


Fig. 1. Four different contact cases in time interval [0, T].

Fig. 1 shows four contact cases observed during time 0 and T, in which a box denotes the period that two nodes are within the communication range of each other. Suppose each message has the same size and can be transmitted within t. We notice that contact frequency fails to distinguish cases (a) and (c) , and average contact duration per contact cannot distinguish cases (a) and (d). Another possible metric is average contact duration per time unit that does well in differentiating case (a) from case (d), but not from (b). Our obervation shows that single-criterion based metric cannot accurately measure the forwarding capacity between two nodes.

4.2 A Hybrid Metric

Motivated by the facts observed from Fig. 1, we design a hybrid metric that measures connection strength by taking both contact frequency and contact duration into consideration. The connection strength $HM_{i,}$ between nodes i and j is defined as:

$$HM_{ij} = \alpha \cdot (1 - e^{-n(T)}) + (1 - \alpha) \cdot \frac{\int_0^T f(t)\, dt}{T}, \tag{4}$$

where $n(T)$ is the contact number between nodes i and j in $[0, T]$, $f(t)$ is a binary value indicating whether i and j are in contact at time t ($f(t)$ = 1) or not ($f(t)$ = 0), and $\alpha \in [0, 1]$ is a tunable scaling parameter according to network feature.

Note that terms $1 - e^{-n(T)}$ and $\int_0^T f(t)\, dt/T$ actually indicate the contact frequency and the average contact duration per time unit, respectively. By tuning parameter α according to network features, the weight of contact frequency and average contact duration can be adjusted. We set high weight on the contact frequency for some DTNs, where the mobile nodes frequently encounter each other but with short duration. For example, dataset Rollernet [22] shows that participants skate together and they frequently encounter and separate. However, for DTNs like Reality [23], where mobile nodes keep a long contact each encounter, the contact duration should be weighted. In some other cases with complicated and irregular encounter patterns, it is difficult to determine the relative importance of contact frequency or average contact time. How the value of α affects the forwarding performance will be further discussed by real-trace driven experiments in Section 7.

Finally, to obtain the social graph on each node, we apply our hybrid connection strength metric HM to the distributed density-based algorithm proposed in [12]. The superiority of the social graph using Eq. (4) over the ones using existing metrics to the routing performance shall be validated by experiments later.

5 Distributed Social-Tie Detection

As we have known, social ties like community structure and bridge node are very helpful to relay node selection in social-based routing protocols. To extract these social ties, we propose a distributed community detection algorithm (DCDA) and a bridge node self-recommending approach based on the obtained social graph.

5.1 Dynamic Community Detection

5.1.1 Community Formation Condition

A community is usually regarded as a group of nodes with more internal connections than external connections in a social graph [18]. To formally quantify such relationship, we introduce intra-connection density d_C^+ and inter-connection density d_C^- of a community C, which are defined as the ratios of the number of internal edges and external edges to all possible internal edges, respectively, i.e.,

$$d_C^+ = \frac{|E_C^+|}{|V_C|(|V_C|-1)/2} \tag{5}$$

and

$$d_C^- = \frac{|E_C^-|}{|V_C|(|V_C|-1)/2}, \tag{6}$$

where E_C^+ and E_C^- denote a set of edges with both endpoints and only one endpoint in C, respectively.

In addition, the number of nodes in a community shall exceed a certain number θ. For example, we usually do not treat two nodes as a community. A recent study [18] [24] shows that the size of communities is usually not less than 4 in mobile social networks. Therefore, in this paper, we assume that all communities shall be with size at least 4, i.e., $\theta = 4$. In summary, a group of nodes can be regarded as a community C if and only if:

$$d_C^+ > d_C^- \text{ and } |C| \geq \theta. \tag{7}$$

From (7), we can see that the community structure would be affected once edge appearance or disappearance happens. Therefore, the community detection algorithm shall be invoked right after the aggregation process, by which an edge may be added into or removed from the social graph.

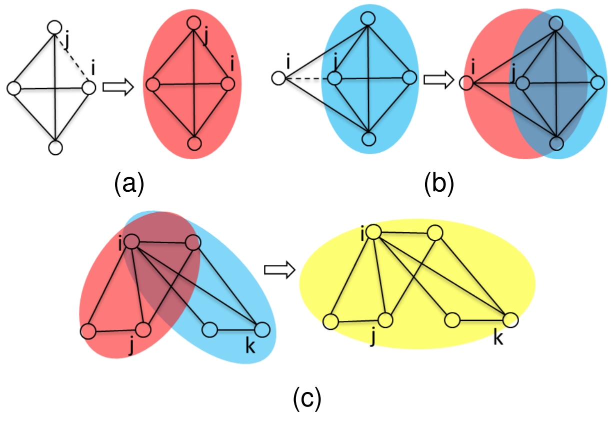


Fig. 2. The Examples of Community Formation

5.1.2 Community Formation

A local social graph is always maintained and updated by each node i during a social graph aggregation process. If a new edge, e.g., e_{ij}, is added in the social graph, the community structure on node i shall be updated by 1) creating a new community or 2) merging two existing communities, as summarized in Algorithm 1 in the supplementary file. Illustrative examples are given in Fig. 2, where communities are shown by shaded regions.

Basic community creation. Whenever a new edge e_{ij} appears, node i always tries to build a new community C_{temp} based on the common neighbors between i and j, no matter whether j is within one or multiple communities (e.g., Fig. 2 (b)) or not (e.g., Fig. 2 (a)). If it satisfies the community formation condition (7) but is not yet formed, C_{temp} will be regarded as a new community and added into the current community set. Otherwise, the appearance of e_{ij} does not affect the existing communities.

Community merging. Sometimes, the new community C_{temp} may share substructures (i.e., overlap) with other existing communities, as shown in Fig. 2(c). In this case, the two communities shall be merged into a bigger one. To this end, after building a new community C_{temp}, Algorithm 1 tries to combine it with its overlapped communities if the merged one still satisfies (7). Finally, the original communities should be removed from the community set.

5.1.3 Community Partition

During the aggregation process, an edge may also disappear from the social graph since a certain density should be kept such that the social graph avoids becoming too dense and homogeneous [12]. The edge removal may incur the partition of certain existing communities. However, existing distributed community detection algorithms [4] [13] simply overlook these events and fail to capture the partition of communities, resulting in inaccuracy in the community structure description. To deal with such events, we propose an algorithm specified in Algorithm 2 as given in the supplementary file.

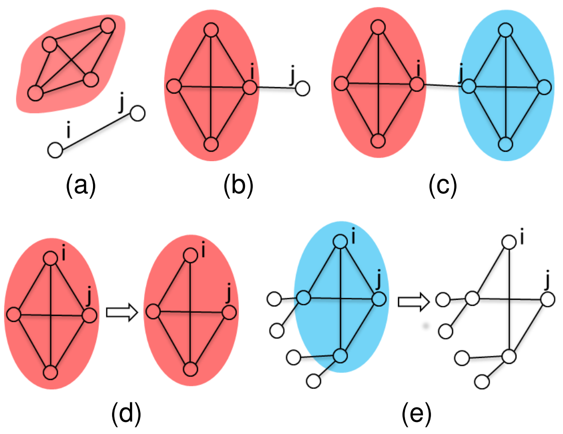


Fig. 3. The Examples of Community Partition

Not all edge disappearance events will result in community partition. For example, if edge e_{ij} does not belong to any community as shown in Fig. 3(a)-(c), its removal will not affect any existing community structure. The existing community structure shall be retained. On the other hand, for the cases (d) and (e) in Fig. 3, where e_{ij} is within a community, the remaining nodes after its deletion should be restructured as follows. If condition (7) still holds, the current community structure shall keep unchanged (e.g., Fig. 3(d)). Otherwise, it will be divided into smaller communities or even disbanded (e.g., Fig. 3(e)). In the latter case, Algorithm 1 shall be applied again to reconstruct the new community structure on node i and all its neighbors in the affected community.

5.2 Bridge Node Identification

Exploring bridge nodes locating between different communities as relay nodes not only helps quick dissemination of messages far away from the original community, but also shortens the transmission hops between source and destination. A prerequisite is to identify the bridge nodes between communities.

5.2.1 Weighted Bridging Centrality

On the measurement of bridge node, it is assumed in (1) that all edges are of the same strength. Under our heterogeneous model, the edges in the social graph are with different weights. The strong edge implies that it could transmit more messages in a limited time, but the un-weighted bridging coefficient defined in (3) ignores such important feature.

To address this problem, we define *weighted bridging coefficient* $\Psi^{'}(i)$ of node i based on our hybrid connection strength metric (4) as follow:

$$\Psi^{'}(i) = \frac{d(i)^{-1}}{\sum_{j \in N_i} d(j)^{-1}}, \tag{8}$$

where $d(i)$ is the weighted degree of node i calculated as

$$d(i) = \sum_{i \neq j \in V_i^t} a(i,j) \times HM_{ij}. \tag{9}$$

Note that $a(i,j)$ in (9) is a binary value indicating whether i connects j in the social graph (i.e, $a(i,j) = 1$) or not (i.e., $a(i,j) = 0$).

In addition, since $\Phi(i)$ in (1) is based on the whole network, we utilize the method presented in [25] to get the approximate betweenness, which correlates to the betweenness based on the entire network. By substituting (8) and the approximate betweenness into (1), we obtain the *weighted bridging centrality*.

The centralized computation of bridging centrality takes time at order of $\Theta(n^3)$, where n is the number of nodes in a network. Our distributed computation achieves much lower time complexity of $\Theta(d_{\max}^2)$, where $d_{\max}$ is the maximum node degree.

5.2.2 Bridge Node Self-Recommending

While the weighted bridging centrality can measure the importance of a node located between communities, it cannot determine whether a node is a bridge node or not. Therefore, we propose a bridge node self-recommending algorithm in this section.

An edge update in a social graph may change the bridging centrality of the involving nodes. In other words, a normal node may become a bridge node or vice versa after the update. Without loss of generality, we consider such updates in a two-hop neighbor network. When two nodes i and j encounter, they shall update their bridging centrality by exchanging information. Based on the updated bridging centrality, Algorithm 3 and Algorithm 4, as given in the supplementary file, will be invoked to deal with bridge node identification and removal, respectively.

Bridge node identification. First of all, if node i is the only node shared by two communities $C \in \mathbb{C}_i^t$ and $C' \in \mathbb{C}_j^t$, and has not yet been detected as a bridge, it will identify itself as a bridge node as shown in lines 3 - 5 of Algorithm 3. Otherwise, node i will recommend itself as a candidate of bridge node between C and C' by sending a *recommendation packet* to each encountered neighbor in C, if either condition below holds as shown in lines 6 - 12 of Algorithm 3, in which $R_i(C, C')$ represents all known bridge candidates across C and C'.

1) Node i has not received any such *recommendation packets* (i.e., $R_i(C, C') = \emptyset$) and its bridging centrality becomes larger.
2) Node i has not been recognized as the bridge node of C and C', i.e., $i \notin B(C, C')$, and the weighted bridging centrality of node i is the highest in all known candidates in $R_i(C, C')$.

Moreover, in the second case, node i considers itself as a bridge node by setting $B_i(C, C') = B_i(C, C') \cup \{i\}$ and $\mathbb{B}_i \leftarrow \mathbb{B}_i \cup \{(C, C')\}$.

Bridge node removal. The weighted bridge centrality of a bridge node varies during the social graph aggregation process. In parallel to the process of bridge node identification, Algorithm 4 is also invoked to determine whether bridge node i would degrade to a normal one when its bridging centrality becomes smaller caused by edge updates in the social graph. For any community pair $C \in \mathbb{C}_i^t$ and $C' \in \mathbb{C}_j^t$, between which i serves as a bridge node, if its weighted bridging centrality is not the highest any more among the candidates in $R_i(C, C')$, node i will cancel its role of bridge node between C and C', and notify each encountered neighbor in C by a *retirement packet*.

Meanwhile, upon the reception of a retirement or recommendation packet, node i shall update R_i by removing the corresponding node or including the recommended one with its associated bridging centrality, respectively.

6 Connection Strength Aware Routing

In this section, we present our distributed CSAR algorithm, which is specified in Algorithm 5 in the supplementary file. It consists of two phases at node i: social-tie detection and routing, when it meets node j. The first phase builds community structure and identifies the bridges between communities by exchanging the social-tie information, as shown in line 1 of Algorithm 5. Then, the updated social ties are utilized to address the problems of *forwarding ordering*, *message copy control*, and *buffer management* in the second phase as follows.

6.1 Forwarding Ordering

When two nodes encounter, since their contact duration is limited and unexpected, it is essential to determine which messages shall be forwarded (i.e., *forwarding candidate set*) and in what order these messages shall be transmitted (i.e., *forwarding order*).

Let $\mathbb{F}_i$ denote the forwarding candidate set at node i, which is constructed as specified in lines 2 - 8 of Algorithm 5. The basic idea is that for each message m carried by node i with a destination $d(m)$, it will be forwarded to node j only if the delivery probability is improved in terms of connection strength, i.e.,

$$\Delta U_{ji}^m \equiv HM_{jd(m)} - HM_{id(m)} > 0. \tag{10}$$

Specifically, only the following two cases will be considered:

1) If the encountered node j is in a community of destination $d(m)$ (line 3 of Algorithm 5), we shall put m in $\mathbb{F}_i$ due to the fact that node j shall frequently encounter its community members.
2) If neither i nor j belongs to any community of the destination $d(m)$ but j is a bridge node (line 5 of Algorithm 5), m shall be also added into $\mathbb{F}_i$. This is because the bridge node would be beneficial to deliver messages to its destination even far way from the current community.

Notice that not all messages in $\mathbb{F}_i$ could be finally forwarded to the encountered node in a short contact duration. In this case, the forwarding order of the messages in $\mathbb{F}_i$ should be carefully considered. Intuitively, messages whose delivery probability can be mostly increased shall be put in the front of the forwarding queue. Therefore, messages in $\mathbb{F}_i$ are sorted and forwarded in a descending order of ΔU_{ji}^m as shown in line 9 of Algorithm 5.

6.2 Message Copy Control

In general, the more copies generated for a message, the higher delivery performance could be achieved. However, this may incur an unaffordable delivery overhead. On the other hand, while limiting the number of message copies conserves consumption of resources (e.g., buffer, energy), it may degrade the delivery performance. Therefore, the message copy control becomes a critical design issue when delivery efficiency is considered.

To balance the tradeoff between the delivery performance and overhead, we propose an *inter-community multi-copy and intra-community single-copy* mechanism. It is applied to deliver each message m in sorted $\mathbb{F}_i$ as shown in lines 10 - 16 of Algorithm 5. Right after forwarding m to the encountered node j, node i will delete m from its buffer if j shares a common community with message m's destination $d(m)$ (line 12), or still keep the message in its local buffer otherwise (line 14). The rationale behind such design is that when message m's current carrier i is far away from the destination node (i.e., node i does not belong to any community of $d(m)$), spreading multiple copies in a network can increase the chance of reaching its destination. On the other hand, once a message reaches its destination's community already, the message is still with a high probability to be finally delivered to its destination even there is only a single copy within the community.

6.3 Buffer Management

During a contact, node i may also receive multiple messages from its encountered node j. Owing to the buffer capacity limitation, we shall consider which messages should be discarded when the buffer overflows. To address this issue, we adopt a greedy approach that the message with the lowest value of $HM_{id(m)}$ is dropped as shown in line 17 of Algorithm 5.

7 Performance Evaluation

7.1 Simulation Setup

In order to evaluate the effectiveness of our proposed hybrid connection strength metrics (i.e., HM) based social graph aggregation, community detection approach (i.e., DCDA) and routing protocol (i.e., CSAR), we conduct intensive simulations on the widely-used DTN simulator ONE [29] using real traces, as summarized in Table 1.

TABLE 1
Characteristics of five mobility datasets

Dataset	Device	Network type	Duration (days)	Granularity (seconds)	No. of nodes	No. of contacts
Infocom06 [26]	iMote	Bluetooth	3	120	98	191,336
Reality [23]	Phone	Bluetooth	246	300	97	54,667
Sassy [27]	T-mote	Bluetooth	79	6.6	27	112,251
Rollernet [22]	iMote	Bluetooth	3(hours)	15	62	60,146
Pmtr [28]	Pmtr	-	19	-	44	11,895

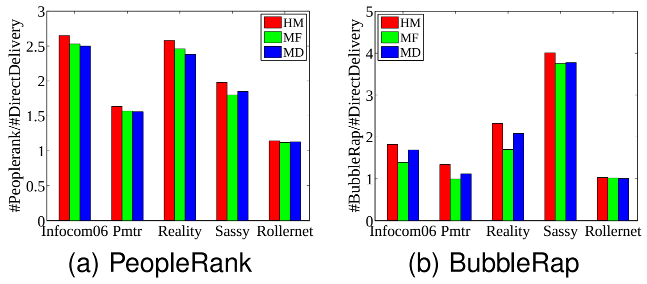


Fig. 4. The delivery performance of routing protocols under various connection strength metrics

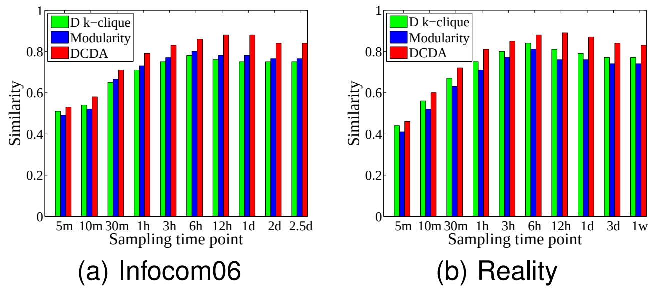


Fig. 5. The similarity of communities detected at different sampling time points.

In our simulations, each message is created with a random interval from 30 to 40 seconds, a source-destination pair randomly chosen from all nodes in the network, a predetermined time-to-live (TTL) representing the delay requirement, and a uniform packet size of 25 KB. Moreover, each node has a buffer size of 3 MB and bandwidth between any two encountered nodes is set as 250 Kbps. For each setting, 40 simulation instances with different random seeds are conducted for statistical confidence.

We employ the following metrics to valuate the efficiency of routing protocols.

- **Delivery Ratio**: the ratio of the number of delivered messages to the number of generated messages.
- **Overhead Ratio**: the average number of forwardings of the same message before it reaches the destination.
- **Average Delay**: the average time for delivering a message to its destination over the network.

7.2 Performance of Connection Strength Metrics

We use MF and MD to denote the metrics purely based on contact-frequency and contact-duration, respectively. To validate the improved quality of aggregated social graphs by our proposed hybrid metrics, we compare the performance of existing social-based routing protocols BubbleRap and PeopleRank under various connection strength metrics HM, MF and MD.

We normalize the delivery ratio of BubbleRap and PeopleRank to the one obtained by direct delivery, which serves as a lower bound on delivery ratio. As shown in Fig. 4, the performance under HM always achieves the best for both BubbleRap and PeopleRank. Our experimental results indicate that even existing unmodified social-based routing protocols can benefit from our hybrid connection strength metric HM. This is attributed to the fact that in general HM provides a more accurate measurement on connection strength that leads to high-quality social graph and superior routing performance.

7.3 Performance of Community Detection Algorithms

In order to evaluate the efficiency of our proposed D-CDA algorithm, we compare it with two state-of-the-art distributed community detection algorithms: distributed k-clique [4] and modularity [13], in terms of *community similarity*, which is defined according to Jaccard index [30] as follows:

$$Sim(C_d, C_c) = \frac{|V_{C_d} \bigcap V_{C_c}|}{|V_{C_d} \bigcup V_{C_c}|}, \tag{11}$$

where C_d and C_c are the communities detected by a distributed algorithm and a centralized algorithm, respectively. We use the centralized k-clique algorithm [12] to generate C_c in our experiments. A larger community similarity indicates that the communities detected by the distributed algorithm are more similar to the ones obtained by a centralized algorithm. We check the average community similarities on datasets Reality and Infocom06 at several sampling points during runtime so as to assess the community evolution.

We find that the similarity by DCDA is always higher than the other two denoted as D k-clique and Modularity in Fig. 5. For example, our DCDA algorithm improves similarity by 17% and 13% compared to distributed k-clique and modularity, respectively, for dataset Infocom06. We attribute this advantage to the unique capability of DCDA in capturing community partition, while both distributed k-clique and modularity only consider community creation.

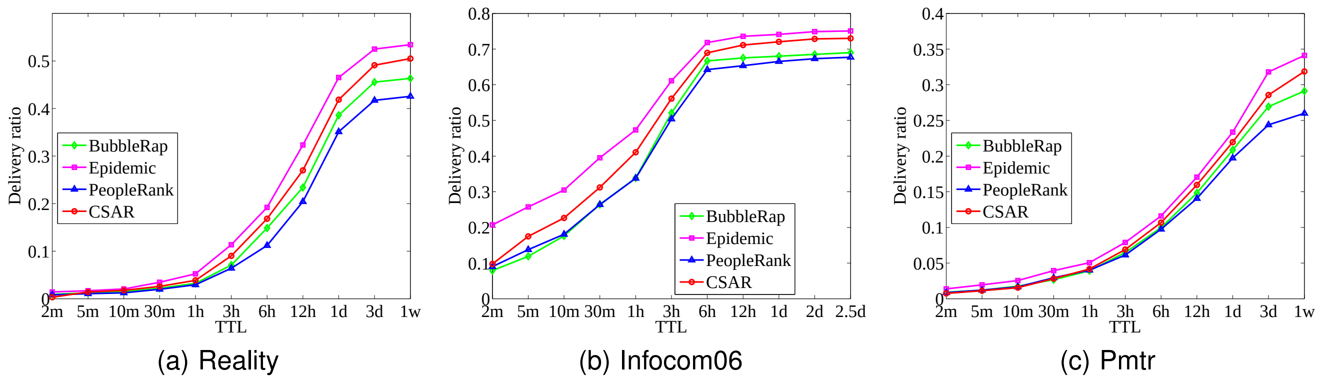


Fig. 6. Comparison of all forwarding algorithms on several datasets in terms of delivery ratio

Another interesting phenomenon observed from Fig. 5 is that all these approaches have a relatively low similarity in the early stage. This is because each node takes time to collect topology information in a distributed way. In addition, the similarity detected by distributed k-clique and modularity decreases faster than that of DCDA after a certain time. As shown in the results from dataset Infocom06, for example, this happens after 24 hours in the DCDA algorithm, but only 6 hours in distributed k-clique and modularity algorithms. The reason behind is that only DCDA adopts a partition detection mechanism to remove those nodes which have already left their previous communities such it achieves higher similarity to the real one.

7.4 Performance of Social-Aware Routing Algorithms

In this section, we evaluate a set of social-aware routing protocols: Epidemic [3], PeopleRank [5], BubbleRap [4], and our proposed CSAR, in which parameter α is set empirically on each dataset. The evaluation on the effect of parameter α can be found in the supplementary file. We present the experimental results based on datasets Infocom06, Reality and Pmtr only since others are similar and thus omitted.

7.4.1 Delivery Ratio

Fig. 6 shows the delivery ratio as a function of TTL. It is clear to see that Epidemic has the highest delivery ratio among all forwarding algorithms since it greedily forwards messages to each encountered node. In addition, we also notice that CSAR outperforms both BubbleRap and PeopleRank. Taking Reality as an example, CSAR outperforms BubbleRap by 27% and PeopleRank by 40% when TTL is three hours. This phenomenon comes from the following reasons.

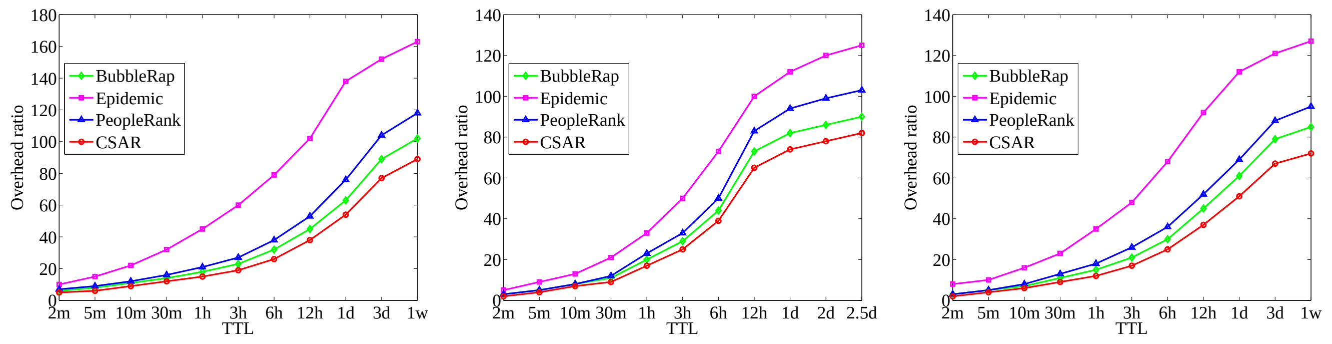
1) First, recall that the performance of social-based forwarding algorithms greatly rely on the social-tie information for relay node selection. The social ties detected in CSAR are based on the high-quality social graph aggregated using our hybrid connection strength metrics defined in (4).
2) We attribute the second reason to the message copy control strategies. Our CSAR algorithm adopts the *inter-community multi-copy* mechanism that spawns copies of a message before it reaches its destination's communities. It significantly increases the message delivery probability. BubbleRap and PeopleRank simply make use of a single replication strategy during the entire routing process.
3) Finally, CSAR considers messages scheduling in a descending order of improved connection strength defined in (10) when multiple ones need to be forwarded at each node, while BubbleRap and PeopleRank simply adopt a random selection strategy each each transmission opportunity.

We also notice from Fig. 6 that the delivery ratio shows as an increasing function of TTL. Furthermore, the maximum delivery ratios vary, i.e., around 50%, 70% and 30% obtained by CSAR in datasets Reality, Infocom06 and Pmtr, respectively, due to the heterogeneity of inter-community contact rates on different datasets. For example, messages destined to the nodes in a different community from its carrier are difficult to be delivered within a given TTL in dataset Pmtr, where inter-community contact rates are low. However, they are more likely to be delivered within the same TTL in dataset Infocom06, where participants with the same interests in a conference session would meet again in other related sessions afterwards.

7.4.2 Delivery Overhead

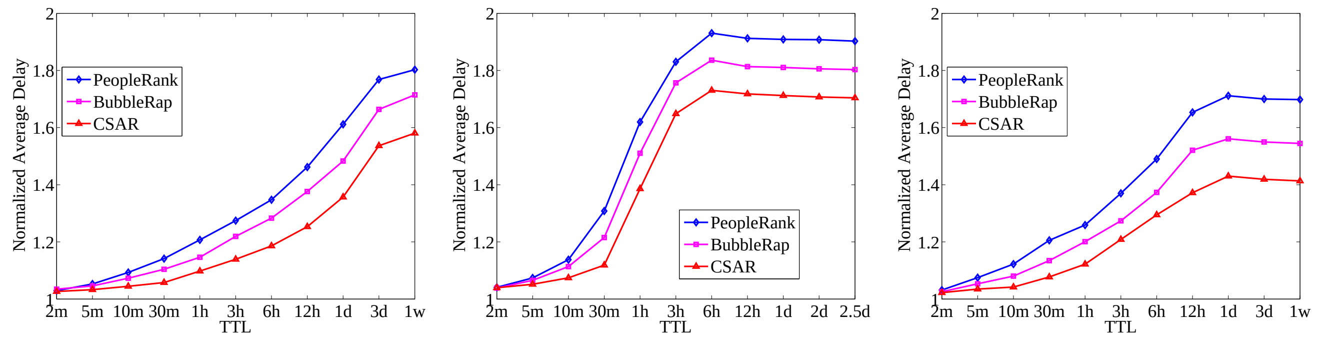
Fig. 7 gives the evaluation results on the overhead ratio of the four routing protocols on different datasets. With no doubt, Epidemic has the highest overhead ratio due to its flooding nature, while CSAR always achieves the lowest overhead ratio in all traces. In Reality, for example, the overhead ratio of CSAR is only 71% and 81% of PeopleRank and BubbleRap, respectively, when TTL is set to a week. Specifically, we have the following observations and conclusions.

1) From the experimental results, we find that messages are forwarded few times by bridge nodes before reaching the destination community, caus-



(a) Reality (b) Infocom06 (c) Pmtr

Fig. 7. Comparisons of all forwarding algorithms on several datasets in terms of overhead ratio



(a) Reality (b) Infocom06 (c) Pmtr

Fig. 8. Comparison of all forwarding algorithms on several datasets in terms of average delay

ing that the multi-copy approach utilized in inter-community communication only brings a small amount of overhead.

2) Both CSAR and BubbleRap utilize community structure to select relay nodes, while PeopleRank overlooks the importance of community. Nodes are more likely to interact with the ones in the same community than others outside the community. To exploit such feature is essential to reduce the number of transmissions.
3) Moreover, community-aware forwarding can prevent a message from flowing outside the destination's community and unnecessary transmissions could be avoided.
4) Finally, CSAR carefully determines the relay candidate set and their forwarding order such that the utility of each transmission opportunity is maximized. In other words, CSAR is able to avoid forwarding messages to those with relatively lower forwarding capabilities. This can improve the transmission efficiency in each transmission opportunity and thus can reduce the overhead significantly. In contrast, both PeopleRank and BubbleRap only make random forwarding for multiple messages.

We also observe that the overhead ratio increases with TTL. This is because larger TTL allows messages to stay longer in a network and thus to obtain more opportunities to be forwarded. Under any value of TTL, the advantage of CSAR can be always observed.

7.4.3 Delivery Delay

Instead of showing the absolute values of delivery delay due to their large span on different TTLs, we present the normalized results in Fig. 8. It defined as the ratio of delay obtained by a social-based routing protocol to the shortest delay achieved by Epidemic because of its flooding nature.

As shown in Fig. 8, CSAR performs better than both BubbleRap and PeopleRank in all cases. Taking Pmtr as an example, the normalized delivery delay of CSAR is 10% and 18% lower than BubbleRap and PeopleRank, respectively, when TTL is set as 12 hours. We attribute its superior performance to the following direct and indirect reasons.

1) The direction reason is that using a multi-copy approach in inter-community transmission shortens delivery delay.
2) Our proposed message scheduling mechasim based on (4) is an indirect reason as explained earlier.
3) Another important indirect reason is that our buffer management scheme drops the messages with the lowest probability to be delivered when the buffer size is insufficient.

Furthermore, we find out from Fig. 8 that the social-based routing protocols perform similarly to Epidemic when the TTL is small but shall deviate when the

TTL increases. This is because when the TTL is small, only the messages whose destinations are close to their sources can be delivered successfully before they expire and hence all routing protocols achieve low delivery delay. However, as TTL becomes larger, the mechanism of various routing protocols starts functioning because many transmission opportunities are available to each message before its expiration. After TTL reaches some value, the average delay converges to a certain value, indicating that the available transmission opportunities become limited under those TTL settings. In summary, the substantial advantage of CSAR is exhibited under any values of TTL.

8 Conclusion

In this paper, we first introduce a new hybrid connection strength metrics in order to aggregate high-quality social graph that can even improve the performance of existing unmodified social-based routing protocols. We then propose the distributed community detection and bridge node identification algorithm DCDA to discover the essential social ties for routing protocol design. It achieves much higher detection accuracy than existing algorithms thanks to its capability of community partition capturing. To fully explore the benefits of our fundamental studies on social graph aggregation and social-tie detection,we propose the routing algorithm CSAR, in which the issues on forwarding ordering, message copy control are buffer management are particularly addressed. Through extensive real-trace driven experiments, the superior efficiency of CSAR is validated as it significantly outperforms existing social-based routing protocols in terms of delivery ratio, overhead ratio and average delay.

References

[1] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of ACM SIGCOMM*, 2003, pp. 27–34.

[2] K. Wei, X. Liang, and K. Xu, "A Survey of Social-Aware Routing Protocols in Delay Tolerant Networks: Applications, Taxonomy and Design-Related Issues," *IEEE Communications Surveys and Tutorials*, no. 99, 2013.

[3] A. Vahdat, D. Becker *et al.*, "Epidemic Routing for Partially Connected Ad hoc Networks," Technical Report CS-200006, Duke University, Tech. Rep., 2000.

[4] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble Rap: Social-based Forwarding in Delay Tolerant Networks." in *Proceedings of ACM MobiHoc*, 2008, pp. 241–250.

[5] A. Mtibaa, M. May, C. Diot, and M. Ammar, "PeopleRank: Social Opportunistic Forwarding," in *Proceedings of IEEE INFOCOM*, 2010, pp. 1–5.

[6] J. Wu and Y. Wang, "Social feature-based Multi-path Routing in Delay Tolerant Networks," in *Proceedings of IEEE INFOCOM*, 2012, pp. 1368–1376.

[7] M. Xiao, J. Wu, C. Liu, and L. Huang, "TOUR: Time-sensitive Opportunistic Utility-based Routing in Delay Tolerant Networks," in *Proceedings of IEEE INFOCOM*, 2013.

[8] C. Liu and J. Wu, "On Multicopy Opportunistic Forwarding Protocols in Nondeterministic Delay Tolerant Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 6, pp. 1121–1128, 2012.

[9] Q. Yuan, I. Cardei, and J. Wu, "An Efficient Prediction-Based Routing in Disruption-Tolerant Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 1, pp. 19–31, 2012.

[10] Y. Li, L. Qiu, D. Jin, L. Su, P. Hui, and L. Zeng, "Contact Duration Aware Evaluation for Content Dissemination Delay in Mobile Social Network," *Wiley Wireless Communications and Mobile Computing*, 2013.

[11] E. M. Daly and M. Haahr, "Social Network Analysis for Routing in Disconnected Delay-tolerant MANETs." in *Proceedings of ACM MobiHoc*, 2007, pp. 32–40.

[12] T. Hossmann, T. Spyropoulos, and F. Legendre, "Know Thy Neighbor: Towards Optimal Mapping of Contacts to Social Graphs for DTN Routing," in *Proceedings of IEEE INFOCOM*, 2010, pp. 1–9.

[13] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft, "Distributed Community Detection in Delay Tolerant Networks," in *Proceedings of ACM/IEEE International Workshop on Mobility in the Evolving Internet Architecture*, 2007, pp. 7:1–7:8.

[14] F. Li and J. Wu, "LocalCom: A Community-based Epidemic Forwarding Scheme in Disruption-tolerant Networks," in *Proceeding of IEEE SECON*, june 2009, pp. 1–9.

[15] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic Networking: Data Forwarding in Disconnected Mobile Ad hoc Networks," *IEEE Communications Magazine*, vol. 44, no. 11, pp. 134–141, Nov. 2006.

[16] P. Hui and J. Crowcroft, "How Small Labels Create Big Improvements," in *Proceedings of IEEE PerCom Workshops*, 2007, pp. 65–70.

[17] T. Nepusz, A. Petróczi, L. Négyessy, and F. Bazsó, "Fuzzy Communities and the Concept of Bridgeness in Complex Networks," *Physical Review E*, vol. 77, no. 1, p. 016107, 2008.

[18] S. Fortunato, "Community detection in graphs," *CoRR*, pp. 75–174, 2009.

[19] W. Gao and G. Cao, "User-centric data dissemination in disruption tolerant networks," in *Proceedings of IEEE INFOCOM*, 2011, pp. 3119–3127.

[20] W. Hwang, T. Kim, M. Ramanathan, and A. Zhang, "Bridging Centrality: Graph Mining from Element Level to Group Level," ser. Proceedings of ACM SIGKDD, 2008, pp. 336–344.

[21] W. Hwang, Y. rae Cho, A. Zhang, and M. Ramanathan, "Bridging Centrality: Identifying Bridging Nodes in Scale-free Networks," Tech. Rep., 2006.

[22] J. Leguay and F. Benbadis, "CRAWDAD data set upmc/rollernet," Downloaded from http://crawdad.cs.dartmouth.edu/upmc/rollernet, Feb. 2009.

[23] N. Eagle and A. Pentland, "Reality Mining: Sensing Complex Social Systems," *Personal and Ubiquitous Computing*, pp. 255–268, 2006.

[24] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Statistical Properties of Community Structure in Large Social and Information Networks," in *Proceedings of ACM WWW*, 2008, pp. 695–704.

[25] P. Pantazopoulos, M. Karaliopoulos, and I. Stavrakakis, "On the Local Approximations of Node Centrality in Internet Router-level Topologies," in *7th International Workshop on Self-Organizing Systems*, 2013.

[26] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD data set cambridge/haggle/imote/infocom2006," Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle/imote/infocom2006, May 2009.

[27] G. Bigwood, D. Rehunathan, M. Bateman, T. Henderson, and S. Bhatti, "data set st andrews/sassy," http://crawdad.cs.dartmouth.edu/standrews/sassy, jun 2011.

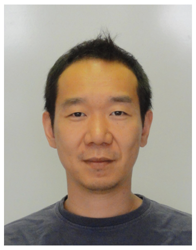
[28] P. Meroni, S. Gaito, E. Pagani, and G. P. Rossi, "data set unimi/pmtr," http://crawdad.cs.dartmouth.edu/unimi/pmtr, dec 2008.

[29] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, 2009.

[30] P. Jaccard, "Étude comparative de la distribution florale dans une portion des Alpes et des Jura," *Bulletin del la Société Vaudoise des Sciences Naturelles*, vol. 37, pp. 547–579, 1901.

Kaimin Wei joined the State Key Lab. of Software Development Environment (SKLSDE) in 2010, and is currently a Ph.D. Student at the Beihang University in Beijing, majoring in computer science. His current research interests are in the areas of wireless mobile networks, delay/disruption tolerant networks, mobile social network and algorithm design.

Deze Zeng received his Ph.D. and M.S. degrees in computer science from University of Aizu, Aizu-Wakamatsu, Japan, in 2013 and 2009, respectively. He is currently a research assistant in University of Aizu, Japan. He received his B.S. degree from School of Computer Science and Technology, Huazhong University of Science and Technology, China in 2007. He is a member of IEEE. His current research interests include cloud computing, networking protocol design and analysis, with a special emphasis on delay-tolerant networks and wireless sensor networks.



Song Guo (M'02-SM'11) received the PhD degree in computer science from the University of Ottawa, Canada in 2005. He is currently a Senior Associate Professor at School of Computer Science and Engineering, the University of Aizu, Japan. His research interests are mainly in the areas of protocol design and performance analysis for reliable, energy-efficient, and cost effective communications in wireless networks. Dr. Guo is an associate editor of the IEEE Transactions on Parallel and Distributed Systems and an editor of Wireless Communications and Mobile Computing. He is a senior member of the IEEE and the ACM.

Ke Xu is a professor at Beihang University, China. He received his B.E., M.E., and Ph.D. degrees from Beihang University in 1993, 1996, and 2000, respectively. His research interests include algorithm and complexity, data mining, and network.