BUBBLE Rap: Social-based Forwarding in Delay Tolerant Networks

Pan Hui, Jon Crowcroft, Eiko Yoneki
University of Cambridge, Computer Laboratory
Cambridge CB3 0FD United Kingdom
[firstname.lastname@cl.cam.ac.uk]

ABSTRACT

In this paper we seek to improve our understanding of human mobility in terms of social structures, and to use these structures in the design of forwarding algorithms for Pocket Switched Networks (PSNs). Taking human mobility traces from the real world, we discover that human interaction is heterogeneous both in terms of hubs (popular individuals) and groups or communities. We propose a social based forwarding algorithm, BUBBLE, which is shown empirically to improve the forwarding efficiency significantly compared to oblivious forwarding schemes and to PROPHET algorithm. We also show how this algorithm can be implemented in a distributed way, which demonstrates that it is applicable in the decentralised environment of PSNs.

Categories and Subject Descriptors

C.2.4 [**Computer Systems Organization**]: Computer Communication Networks—*Distributed Systems*; I.6 [**Computing Methodologies**]: Simulation and Modeling

General Terms

Measurement, Experimentation, Algorithms

Keywords

Social Network, Forwarding, Delay Tolerant Network, Pocket Switched Network, Community, Centrality

1. INTRODUCTION

We envision a future in which a multitude of devices carried by people are dynamically networked. We aim to build PSN [10]: a type of Delay Tolerant Networks (DTN) [6] for such environments. A PSN uses contact opportunities to allow humans to communicate without network infrastructure.[1] We require an efficient data forwarding mechanism over the temporal graph of the PSN [15], that copes with dynamical, repeated disconnection and re-wiring. End-to-end delivery through traditional routing algorithms is not applicable.

Many MANET and some DTN routing algorithms [14] [19] provide forwarding by building and updating routing tables whenever mobility occurs. We believe this approach is not cost effective for a PSN, since mobility is often unpredictable, and topology changes can be rapid. Rather than exchange much control traffic to create unreliable routing structures, we prefer to search for some characteristics of the network which are less volatile than mobility. A PSN is formed by people. Those people's social relationships may vary much more slowly than the topology, and therefore can be used for better forwarding decisions. Furthermore, if we can detect these social mobility patterns online in a decentralised way, we can put the algorithms into practical applications.

In this paper, we focus on two specific aspects of society: community and centrality. *Community* is an important attribute of PSNs. Cooperation binds, but also divides human society into communities. Human society is structured. Within a community, some people are more popular, and interact with more people than others (i.e.have high *centrality*); we call them hubs. Popularity ranking is one aspect of the population. For an ecological community, the idea of correlated interaction means that an organism of a given type is more likely to interact with another organism of the same type than with a randomly chosen member of the population [27]. This correlated interaction concept also applies to human, so we can exploit this kind of community information to select forwarding paths.

Methodologically, community detection [25] [4] can help us to understand local community structure in both offline mobile trace analysis and online applications, and is therefore helpful in designing good strategies for information dissemination. Freeman [8] defined several centrality metrics to measure the importance of a node in a network. Betweenness centrality measures the number of times a node falls on the shortest path between two other nodes. This concept is also valid in a DTN. In a PSN, it can represent the importance of a node as a potential traffic relay for other nodes in the

[1]Regarding the motivations of PSN, there is a huge amount of untapped resources in portable networked devices such as laptops, PDAs and mobile phones, including local wireless bandwidth (e.g. 802.11 and Bluetooth), storage capacity, CPU power, and multimedia data. These resources should be utilised. Furthermore, the communication between users is not always necessarily to pass through the Internet. According to a questionnaire survey amount 70 participants in the Computer Laboratory University of Cambridge, around 50% of their email exchanges are among people they met daily. Another motivation is that the information provide by the Internet may not best satisfy the interest of the local users, for example an user may be more interested in a video clip of his friend, Britney Spears, instead of the MTV of the singer Britney Spears that is usually what Google search will return to you. Empirical result about social search are also observed by other researchers [21]. In this aspect, PSN unleashes the power of local, social and community search and communication.

MobiHoc'08, May 26–30, 2008, Hong Kong SAR, China.

Experimental data set	Infocom05	Hong-Kong	Cambridge	Infocom06	Reality
Device	iMote	iMote	iMote	iMote	Phone
Network type	Bluetooth	Bluetooth	Bluetooth	Bluetooth	Bluetooth
Duration (days)	3	5	11	3	246
Granularity (seconds)	120	120	600	120	300
Number of Experimental Devices	41	37	54	98	97
Number of internal contacts	22,459	560	10,873	191,336	54,667
Average # Contacts/pair/day	4.6	0.084	0.345	6.7	0.024

Table 1: Characteristics of the five experimental data sets

system. The main contributions of this paper are to answer these questions:

1. How does the variation in node popularity help us to forward in a PSN?
2. Are communities of nodes detectable in PSN traces?
3. How well does social based forwarding work, and how does it compare to other forwarding schemes in a real (emulated) environment?
4. Can we devise a fully decentralised way for such schemes to operate?

Quick answers to the above questions, we evaluate the impact of community and centrality on forwarding, and propose a hybrid algorithm, BUBBLE, that selects high centrality nodes and community members of destination as relays. We demonstrate a significant improvement in forwarding efficiency over oblivious forwarding and the PROPHET algorithm [19], which uses patterns of movement, rather than the longer term social relationships that our scheme infers. In a PSN, there may be no a priori information. By definition, we are also in a decentralised world without access to infrastructure. Therefore the distributed detection and dissemination of node popularity and node communities, and the use of these for forwarding decisions are crucial. We verify that this is not only possible, but works well in terms of packet delivery performance and efficiency compared to prior schemes.

The rest of this paper is structured around the theme of social based forwarding as follows. In Section 2, we introduce the experimental datasets used in the paper. We then introduce the general ideas of BUBBLE in Section 3, followed by the study of centralised community detection algorithms in Section 4. We show the possibility of a distributed implementation for BUBBLE in Section 5. In Section 6, we empirically evaluate the BUBBLE algorithm. Related work is described in Section 7. Finally we conclude the paper with a brief discussion and suggested future work.

2. EXPERIMENTAL DATASETS

In this paper, we use three experimental datasets gathered by the Haggle Project [2] over two years, referred to as *HongKong*, *Cambridge*, *Infocom06*; one dataset from the MIT Reality Mining Project [5], referred to as *Reality*. Previously, the characteristics of these datasets such as inter-contact and contact distribution have been explored in several studies [2] [10] [18], to which we refer the reader for further background information. We believe these four datasets cover a rich diversity of environments from busy metropolitan city (*HongKong*) to quite university town (*Cambridge*), with an experimental period from several days (*Infocom06*) to almost one year (*Reality*).

- In *Infocom05*, the devices were distributed to approximately fifty students attending the Infocom student workshop. Participants belong to different social communities (depending on their country of origin, research topic, etc.). However, they all attended the same event for 4 consecutive days and most of them stayed in the same hotel and attended the same sections (note, though, that Infocom is a multi-track conference).

- In *Hong-Kong*, the people carrying the wireless devices were chosen independently in a Hong-Kong bar, to avoid any particular social relationship between them. These people have been invited to come back to the same bar after a week. They are unlikely to see each other during the experiment.

- In *Cambridge*, the iMotes were distributed mainly to two groups of students from University of Cambridge Computer Laboratory, specifically undergraduate year1 and year2 students, and also some PhD and Masters students. This dataset covers 11 days.

- In *Infocom06*, the scenario was very similar to *Infocom05* except that the scale is larger, with 80 participants. Participants were selected so that 34 out of 80 form 4 subgroups by academic affiliations.

- In *Reality*, 100 smart phones were deployed to students and staff at MIT over a period of 9 months. These phones were running software that logged contacts with other Bluetooth enabled devices by doing Bluetooth device discovery every five minutes.

The five experiments are summarised in Table 1.

3. BUBBLE RAP FORWARDING

In previous work, Hui *et al.* introduced the LABEL scheme [11]. Each node is assumed to have a label that informs other nodes of its affiliation; next-hop nodes are selected if they belong to the same affiliation (same label) as the destination. It was demonstrated that LABEL significantly improves forwarding efficiency over oblivious forwarding using their one dataset (Infocom06). This is a beginning of social based forwarding in PSN, but without a concise concept of community and lack of mechanisms to move messages away from the source when the destinations are socially far away(such as *Reality* [3]).

Here we propose the BUBBLE algorithm, with the intention of bringing in a concise concept of community into PSN forwarding to achieve significant improvement of forwarding efficiency. BUBBLE combines the knowledge of community structure with the knowledge of node centrality to make forwarding decisions. There are two intuitions behind this algorithm. Firstly, people have varying roles and popularities in society, and these should be true also in the network – the first part of the forwarding strategy is to forward

[2]http://www.haggleproject.org

[3]We show the details of these datasets in the following paragraphs of this section.

messages to nodes which are more popular than the current node. Secondly, people form communities in their social lives, and this should also be observed in the network layer – hence the second part of the forwarding strategy is to identify the members of destination communities, and to use them as relays. Together, we call this BUBBLE forwarding.

For this algorithm, we make two assumptions:

- Each node belongs to at least one community. Here we allow single node communities to exist.
- Each node has a global ranking (i.e.global centrality) across the whole system, and also a local ranking within its local community. It may also belong to multiple communities and hence may have multiple local rankings.

Forwarding is carried out as follows. If a node has a message destined for another node, this node first *bubbles* the message up the hierarchical ranking tree using the global ranking, until it reaches a node which is in the same community as the destination node. Then the local ranking system is used instead of the global ranking, and the message continues to bubble up through the local ranking tree until the destination is reached or the message expires. This method does not require every node to know the ranking of all other nodes in the system, but just to be able to compare ranking with the node encountered, and to push the message using a greedy approach. In order to reduce cost, we also require that whenever a message is delivered to the community, the original carrier can delete this message from its buffer to prevent further dissemination. This assumes that the community member can deliver this message. We call this algorithm BUBBLE, using the metaphor of *bubble* for a community.

Algorithm 1: BUBBLE RAP

```
begin
    foreach EncounteredNode_i do
        if (LabelOf(currentNode) == LabelOf(destination)) then
            if (LabelOf(EncounteredNode_i) ==
                                        LabelOf(destination))
                                    and
            (LocalRankOf(EncounteredNode_i) >
                                        LocalRankOf(currentNode))
            then
                EncounteredNode_i.addMessageToBuffer(message)
        else
            if (LabelOf(EncounteredNode_i) ==
                                        LabelOf(destination))
                                    or
            (GlobalRankOf(EncounteredNode_i) >
                                        GlobalRankOf(currentNode))
            then
                EncounteredNode_i.addMessageToBuffer(message)
end
```

The forwarding process fits our intuition and is taken from real life experiences. First you try to forward the message via surrounding people more popular than you, and then you bubble it up to well-known popular people in the wider-community, such as a postman. When the postman meets a member of the destination community, the message will be passed to that community. The first community member who receives the message will try to identify more popular members within the community, and bubble the message up again within the local hierarchy, until the message reaches a very popular member, or the destination itself, or the message expires. Figure 1 illustrates the BUBBLE algorithm and Algorithm 1 summarise the operations in a flat community (not hierarchical [4])

[4]We will discuss the hierarchical structures in the conclusion section.

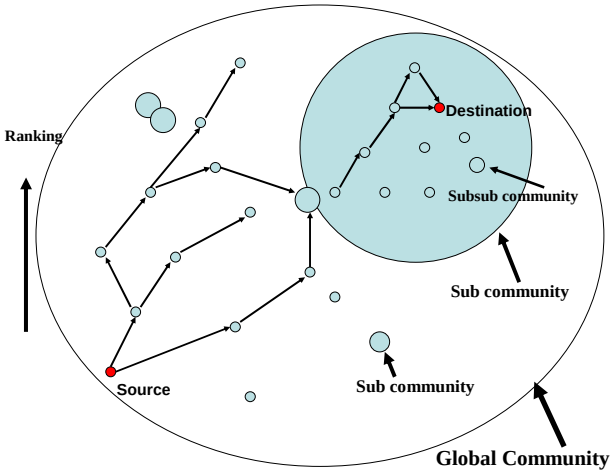

Figure 1: Illustration of the BUBBLE algorithm

space.

In the rest of this paper, we evaluate BUBBLE to confirm our intuition that social based forwarding in general, and BUBBLE specifically, form a viable and effective approach in PSNs, and answer the four questions posed in the introduction.

We answer the first question by looking at the human heterogeneity in the dataset. To calculate the individual centrality value for each node, we take an numerical approach. First we carry out a large number of emulations of unlimited flooding with different uniformly distributed traffic patterns created using the *HaggleSim* emulator [11], which can replay the collected mobility traces and emulate different forwarding strategies for every contact event. Then we count the number of times a node acts as a relay for other nodes on all the shortest delay deliveries. Here the shortest delay delivery refers to the case when the same message is delivered to the destination through different paths, and we only count the delivery with the shortest delay. We call this number the betweenness centrality of this node in this temporal graph [5]. Of course, we can normalise it to the highest value found. Here we use unlimited flooding since it can explore the largest range of delivery alternatives with the shortest delay. This definition captures the spirit of the Freeman centrality [8].

Initially, we only consider a homogeneous communications pattern, in the sense that every destination is equally likely, and we do not weigh the traffic matrix by locality. We then calculate the global centrality value for the whole homogeneous system. Later, we will analyse the heterogeneous system, once we have understood the community structure.

Figure 2 shows the number of times a node falls on the shortest paths between all other node pairs. We can simply treat this as the centrality of a node in the system. We observed a very wide heterogeneity in each experiment. This clearly shows that there is a small number of nodes which have extremely high relaying ability, and a large number of nodes have moderate or low centrality values, across all experiments. The 30, 70 percentiles and the means of normalised individual node centrality are shown in Table 2, which tell the heterogeneity of each system.

This matches well with our intuition of human heterogeneity. People differ in their popularity. For instance, a salesperson or politician interacts with many others, making themselves highly-ranked nodes in our graph, compared to (say) the average computer

[5]We will show in Section 5 how to approximate it in a distributed way.

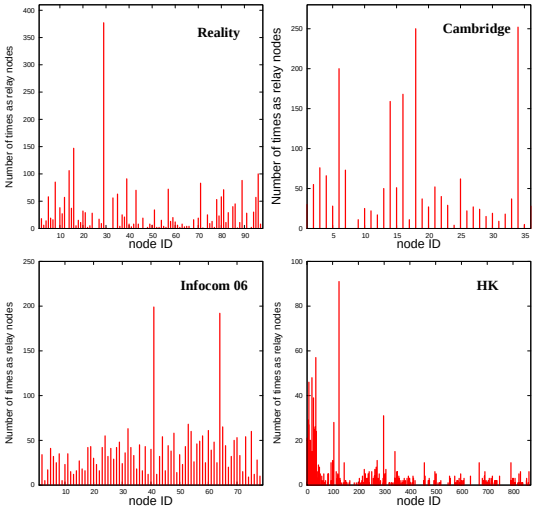

Figure 2: Frequency of nodes as relays

scientist. Homogeneity might favour different forwarding strategies for PSNs. In contrast, we want to employ heterogeneous popularity to help designing more efficient forwarding strategies: we prefer to choose popular hubs as relays rather than unpopular ones.

Dataset	30 percentile	70 percentile	Mean
HongKong	0.000	0.000	0.017
Reality	0.005	0.050	0.070
Infocom06	0.121	0.221	0.188
Cambridge	0.052	0.194	0.220

Table 2: Normalised node centrality across experiments

4. INFERRING HUMAN COMMUNITIES

A social network consists of a set of people forming socially meaningful relationships, where prominent patterns or information flow are observed. In PSN, social networks could map to computer networks since people carry the computer devices. To answer the second question we need community detection algorithms. In this section, we introduce and evaluate two centralised community detection algorithms: K-CLIQUE by Palla *et al.* and weighted network analysis (WNA) by Newman [28] [24]. We use these two centralised algorithm to uncover the community structures in the mobile traces, and we believe our evaluation of these algorithms is also beneficial for the future traces study by the mobile research community.

Many centralised community detection methods have been proposed and examined in the literature (see the recent review papers by Newman [25] and Danon *et al.* [4]). The criteria we use to select a centralised detection method are the ability to uncover overlapping communities, and a high degree of automation (low manual involvement). In real human societies, one person may belong to multiple communities and hence it is important to uncover this feature when we study human networks. K-CLIQUE method can satisfy this requirement, but was designed for binary graphs, thus we must threshold the edges of the contact graphs in our mobility traces to use this method and it is difficult to choose an optimum threshold manually [28]. On the other hand, (WNA) can work on weighted graphs directly, and doesn't need thresholding, but it cannot detect overlapping communities [24]. Thus we chose to use both K-CLIQUE and WNA; they have favourable features and can compliment each other.

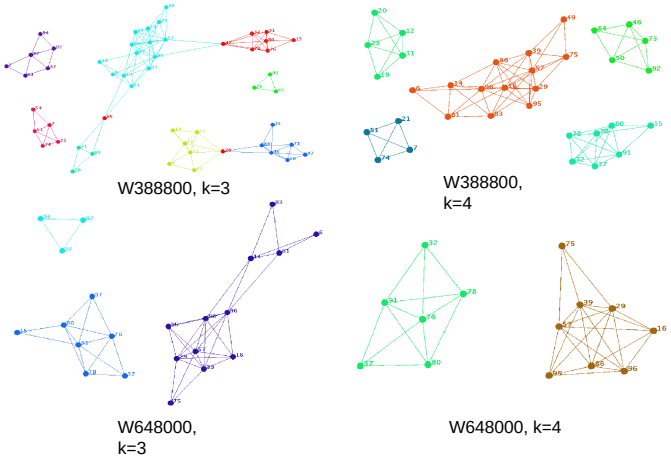

Figure 3: Communities based on contact durations with weight threshold = 388800s (4.5days), 648000s (7.5days) and k=3,4 (*Reality*)

4.1 K-CLIQUE Community Detection

Palla *et al.* define a k-clique community as a union of all k-cliques (complete subgraphs of size k) that can be reached from each other through a series of adjacent k-cliques, where two k-cliques are said to be adjacent if they share $k-1$ nodes. As k is increased, the k-clique communities shrink, but on the other hand become more cohesive since their member nodes have to be part of at least one k-clique. We have applied this on all the datasets above. Here we take *Reality* as an example, since it contains a reasonably large number of nodes, and lasts for a long period of time. Out of 100 experimental participants, 75 are either students or faculty in the MIT Media Laboratory, while the remaining 25 are incoming students at the adjacent MIT Sloan business school. Of the 75 users at the Media lab, 20 are incoming masters students and 5 are incoming MIT freshmen.

First we look at communities detected by using a contact threshold of 388,800 seconds or 4.5 days on the 9 months *Reality* dataset. The threshold was obtained from assuming 3 lectures per week;with 4 weeks per month and a total trace duration of 9 months(2% of the total links are taken into consideration). Research students in the same office may stay together all day, so their contact duration threshold could be very large. For students attending lectures, this estimation can be reasonable. Using a looser threshold still detects the links with much stronger fit. We observe 8 communities of size (16,7,7,7,6,5,4,3) when $k=3$. When $k=4$, the 3-clique community is eliminated and other communities shrink or are eliminated, and only 5 communities of size (13,7,5,5,4) left. All of these 5 communities are disjoint. When $k=5$, 3 communities of size (9,6,5) remains, the size-9 one and the size-5 one are split from the 13-sized one in the 4-clique case. Moving to $k=6$ and $k=7$, there are 2 communities and 1 community respectively. We are also interested in knowing about small groups which are tightly knit. We set a strict threshold of 648,000 seconds, that is on average 1 hour per weekday, 4 weeks per month, and for a total of 9 months. Around 1% of the links are taken into account for the community detection. When $k=3$, there are three disjoint communities of size (12,7,3). When $k=4$, there are only two communities left of size (8,6). Figure 3 shows the 3-clique and 4-clique communities of 648,000 seconds threshold with its counter parts of 388,800 seconds. A single 7-clique community remains in $k=5$ and $k=6$ cases, this 7-clique community is the same as in

the 388,800 second case. These 7 people could be people from a same research group, they know each other and have long contact with each other.

4.2 Weighted Network Analysis

In this section, we implement and apply Newman's weighted network analysis (WNA) for our data analysis [24]. [6] Our contribution is also the extension of the unweighted *modularity* proposed in [26] to a weighted version, and use this as a measurement of the fitness of the communities it detects.

For each community partitioning of a network, one can compute the corresponding modularity value using the following definition of *modularity* (Q):

$$Q = \sum_{vw} \left[\frac{A_{vw}}{2m} - \frac{k_v k_w}{(2m)^2} \right] \delta(c_v, c_w) \quad (1)$$

where A_{vw} is the value of the weight of the edge between vertices v and w, if such an edge exists, and 0 otherwise; the δ-function $\delta(i,j)$ is 1 if $i = j$ and 0 otherwise; $m = \frac{1}{2}\sum_{vw} A_{vw}$; k_v is the degree of vertex v defined as $\sum_w A_{vw}$; and c_i denotes the community of which vertex i belongs to. Therefore the term in the formula $\frac{\sum_{vw} A_{vw}}{2m}\delta(c_v,c_w)$ is equal to $\frac{\sum_{vw} A_{vw}\delta(c_v,c_w)}{\sum_{vw} A_{vw}}$, which is the fraction of the edges that fall within communities. *Modularity* is defined as the difference between this fraction and, the fraction of the edges that would be expected to fall within the communities if the edges were assigned randomly but keeping the degrees of the vertices unchanged. The algorithm is essentially a genetic algorithm, using the modularity as the measurement of fitness. Instead of testing on some mutations of the current best solutions, it enumerates all possible merges of any two communities in the current solution, evaluates the relative fitness of the resulting merges, and chooses the best solution as the seed for the next iteration.

Table 3 summarises the communities detected by applying WNA on the four datasets. According to Newman [24], nonzero Q values indicate deviations from randomness; values around 0.3 or more usually indicate good divisions. For the *Infocom06* case, the Q_{max} value is low; this indicates that the community partition is not very good in this case. This also agrees with the fact that in a conference the community boundary becomes blurred. For the *Reality* case, the Q value is high; this reflects the more diverse campus environment. For the *Cambridge* data, the two groups spound by WNA is exactly matched the two groups (1st year and 2nd year) of students selected for the experiment.

Dataset	Info06	Camb	Reality	HK
Q_{max}	0.2280	0.4227	0.5682	0.6439
Max. Community Size	13	18	23	139
No. Communities	4	2	8	19
Avg. Community Size	8.000	16.500	9.875	45.684
No. Community Nodes	32	33	73	868
Total No. of Nodes	78	36	97	868

Table 3: Communities detected from the four datasets

These centralised community detection algorithms give us rich information about the human social clustering and are useful for offline data analysis on mobility traces collected. We can use them to explore structures in the data and hence design useful forwarding strategies, security measures, and killer applications.

[6]During the implementation, we found two different interpretations of the algorithm in the paper. We believe that we have chosen the correct one after the confirmation of the author.

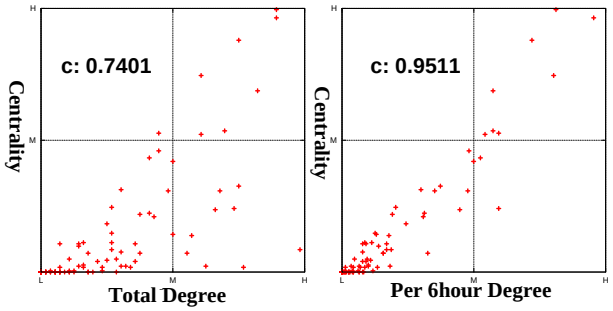

Figure 4: Correlation of rank with total degree and rank with unit time degree (*Reality*)

5. DISTRIBUTED BUBBLE RAP

For practical applications, we want to look further into how BUBBLE can be implemented in a distributed way. To achieve this, each device should be able to detect its own community and calculate its centrality values. Hui *et al.* proposed three algorithms, named SIMPLE, K-CLIQUE and MODULARITY, for distributed community detection, and they proved that the detecting accuracy can be up to 85% of the centralised K-CLIQUE algorithm [12]. Here we introduce our distributed BUBBLE algorithm, DiBuBB, which uses the methods of Hui *et al.* [12] to detect communities and uses our approximation method to calculate individual centrality value in a decentralised manner.

From trace analysis, we found that the total degree (unique nodes seen by a node throughout the experiment period) is not a good approximation of the node centrality. Instead the degree per unit time (for example the number of unique nodes seen per 6 hours [7]) and the node centrality have a high correlation value. We can see from Figure 4 that some nodes with a high total degree are still not good carriers. It also shows that the 6-hour degree is well correlated to the centrality value, with correlation coefficient as high as 0.9511. Therefore the number of people you know is less important matter too much, but how frequently you interact with these people does matter. For convenience of notation, we refer to this average unit-time degree as DEGREE.

However, the average unit-time degree calculated throughout the whole experimental period is still difficult for each node to calculate individually. We then consider the degree for previous unit-time slot (we call this the slot window) such that when two nodes meet each other, they compare how many unique nodes they have met in the previous unit-time slot (e.g. 6 hours). We call this approach the single window (S-Window). Another approach is to calculate the average value on all previous windows, such as from yesterday to now, then calculate the average degree for every 6 hours. We call this approach the cumulative window (C-Window). This technique is similar to exponential smoothing [31], which we will investigate in further work.

We will further show in Section 6 that DEGREE, S-Window, and C-Window can approximate the pre-calculated centrality quite well and the centrality measured in the past can be used as future predictor. Here we first specify DiBuBB as an algorithm, which uses the distributed K-CLIQUE algorithm to detect local commmunity and C-Window to approximate its own global and local centrality values. Besides that, it operate exactly like BUBBLE.

[7]We chose 6 hours here based on our intuition that daily life is divided into 4 main periods – morning, afternoon, evening and night – each almost 6 hours. But we want to examine the impact of this period in the future.

6. RESULTS AND EVALUATIONS

In order to evaluate different forwarding algorithms, we use the same *HaggleSim* emulator as we used in Section 3. The original trace files are divided into discrete sequential contact events, and they are fed into the emulator as inputs. For every discrete encounter event, the emulator makes a forwarding decision based on the forwarding algorithm under study.

For each emulation in this paper, 1000 messages are created, uniformly sourced between all node pairs. Each emulation is repeated 20 times with different random seeds for statistical confidence. For all the emulations we have conducted for this work, we have measured the following metrics and for all the metrics, we compute the 95th percentile using t-distribution.

Delivery ratio: The proportion of messages that have been delivered out of the total unique messages created.
Delivery cost: The total number of messages (include duplicates) transmitted across the air. To normalize this, we divide it by the total number of unique messages created.
Hop-distribution for deliveries: The distribution of the number of hops needed for all the deliveries. This reveals the social distance between sources and destinations.

We compare our algorithms against the following five benchmark algorithms [8]

WAIT: Hold on to a message until the sender encounters the recipient directly, which represents the lower bound for delivery and cost.
FLOOD: Messages are flooded throughout the entire system, which represents the upper bound for delivery and cost.
MCP: Multiple-Copy-Multiple-Hop. Multiple Copies are sent subject to a time-to-live hop count limit on the propagation of messages. By exhaustive emulations, a 4-copy-4-hop MCP scheme is found to be most cost effective scheme in term of delivery ratio and cost for all naive schemes among all the datasets except the *HongKong* data. Hence for fair comparison, we evaluate our algorithms against the 4-copy-4-hop MCP scheme in most of the cases.
LABEL: A social based forwarding algorithm introduced by Hui *et. al* [11]. Messages are only forwarded to the nodes in the same community (i.e.with the same label) as the destination.
PROPHET: A standard non-oblivious benchmark that has been evaluated against several previous works[19]. It calculates the delivery predictability at each node for each destination by using history of encounters and transitivity. A message is forwarded to a node if it has higher delivery predictability than the current node for that particular destination.

A complete evaluation of DiBuBB need to analyse the effect of dynamic Familiar Set thresholds, the evolution of the communities detected at different times, and also the effect of aging of the contacts(see Section 5.3 of [12]), which would be out of the length of this paper. In this paper, we focus on the evaluation using centralised K-clique communities and pre-calculated centralities. However we also show the evaluation results of DEGREE, S-Window, C-Window, and predictability of centrality as part of divide-and-conquer solution of DiBuBB(DiBuBB consists of two modules: distributed community detection and distributed centrality approximation). We will put the complete analysis of DiBuBB in another paper as a follow up.

6.1 Two-Community Case

In order to make the analysis more systematic, we start with the two-community case. We use the *Cambridge* dataset for this study. The *Cambridge* data can clearly be divided into two communities - the undergraduate year 1 (Group A) and year 2 (Group B) groups, both by experimental design, and as confirmed by our community detection algorithms.

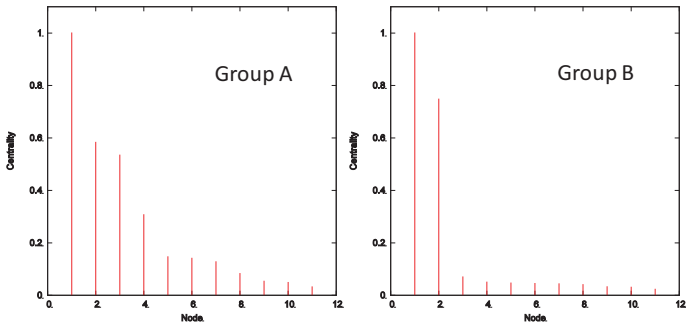

Figure 5: Node centrality in 2 groups (*Cambridge*)

First we look at the simplest case, for the centrality of nodes within each group. In this case, the traffic is created only between members of the same community and only members in the same community are chosen as relays for messages. We can see clearly from Figure 5 that the centrality of each node is different inside a community. In Group B, there are two nodes which are very popular, and relayed most of the traffic. All the other nodes have very low centrality values. Forwarding messages to the popular nodes would make delivery more cost effective for messages within the same community.

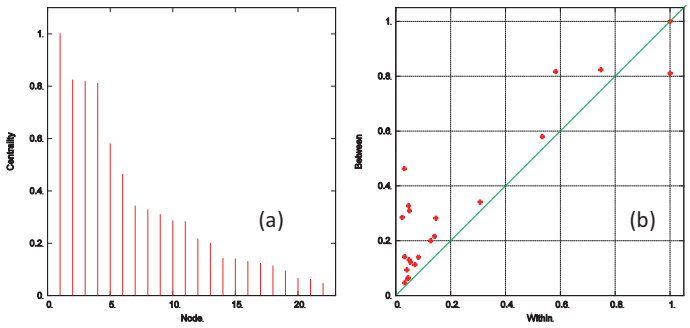

Figure 6: Inter-group centrality(left) and correlation between intra and inter-group centrality(right), (*Cambridge*)

Then we consider traffic which is created from one group and only destined for members in another group. To eliminate other outside factors, we use only members from these two groups as relays. Figure 6(a) shows the individual node centrality when traffic is created from one group to another. Figure 6(b) shows the correlation of node centrality within an individual group and inter-group centrality. We can see that points lie more or less around the diagonal line. This means that the inter- and intra- group centralities are quite well correlated. Active nodes in a group are also active nodes for inter-group communication. There are some points on the left hand side of the graph which have very low intra-group centrality but moderate inter-group centrality. These are nodes which move across groups. They are not important for intra-group communication but can perform certainly well when we need to move traffic from one group to another. We can see from Figure 7 that BUBBLE achieves almost the same delivery success rate as the 4-copy-4-hop MCP but with only 45% of its cost. These two groups share the same education building and usually would overlap with each other

[8]In some graphs we only show the performance of optimised MCP, BUBBLE, and LABEL, in order to focus on the comparison among them.

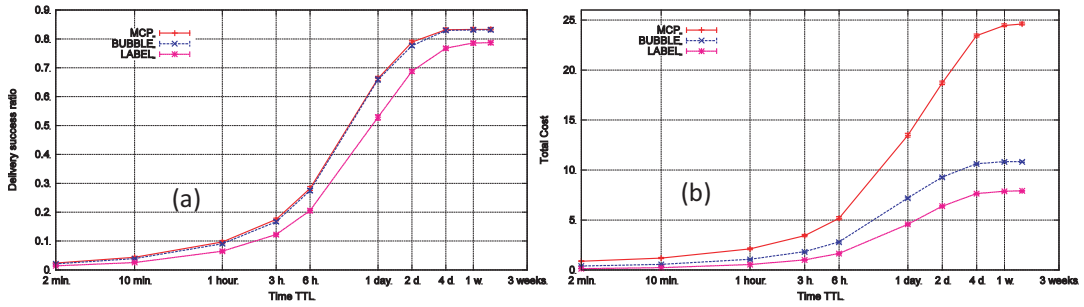

Figure 7: Comparisons of several algorithms on *Cambridge* dataset, delivery and cost.

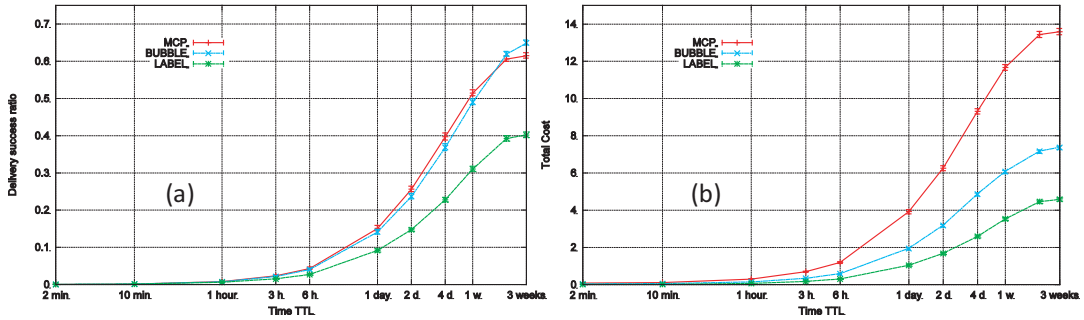

Figure 8: Comparisons of several algorithms on *Reality* dataset, single group.

in the common area, so LABEL behaves quite well in this environment but still has a delivery ratio around 20% to 30% lower than BUBBLE.

6.2 Multiple-Community Case

To study the multiple-community case, we use the *Reality* dataset. To evaluate the forwarding algorithm, we extract a 3 week session during term time from the whole 9 month dataset. Emulations were run over this dataset with uniformly generated traffic. There is a total 8 groups within the whole dataset. We observed that within each individual group, the node centralities demonstrate diversity similar to the *Cambridge* case. In order to make our study easier, we first isolate just one group, consisting of 16 nodes. In this case, all the nodes in the system create traffic for members of this group. We can see from Figure 8(a) that BUBBLE performs very similarly to MCP most of the time in the single-group case, and even outperform MCP when the time TTL is set to be longer than 1 week. From Figure 8(b), we can see that BUBBLE only has 55% of the cost of MCP. We can say that the BUBBLE algorithms are much more cost effective than MCP, with high delivery ratio and low delivery cost.

After the single-group case, we start looking at the inter-group communication for multiple-group. We want to find the upper cost bound for BUBBLE algorithm, so we do not consider local ranking; messages can now be sent to all members in the group. We do not implement the mechanism to remove the original message after it has been delivered to the group member, so the cost here will represent an upper bound for BUBBLE type algorithms.

From Figure 9(a) and Figure 9(b), we can see that of course flooding achieves the best for delivery ratio, but the cost is 2.5 times that of MCP, and 5 times that of BUBBLE. BUBBLE is very close in performance to MCP in the multiple-group case as well, and even outperforms it when the time TTL of the messages is allowed to be larger than 2 weeks. However, the cost is only 50% that of MCP.

Regarding our critics about LABEL in Section 3, we can observe from Figure 9 that LABEL only achieves around 55% of the delivery ratio of the MCP strategy and only 45% of the flooding delivery although the cost is also much lower. However it is not an ideal scenario for LABEL. In this environment, people do not mix as well as in a conference [11]. A person in one group may not meet members in another group so often, waiting to meet a member of the destination group before transmitting is not effective. Figure 10 shows the

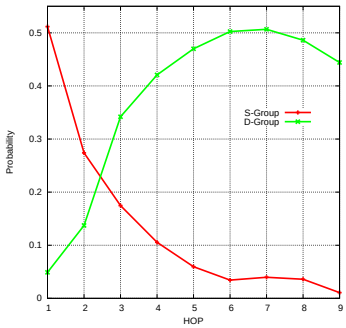

Figure 10: Correlation of nth-hop nodes with the source group and destination group (*Reality*)

correlation of the nth-hop relay nodes to the source and destination groups (S-Group and D-Group) for the messages on all the shortest paths, that is the percentage of the nth-hop relay nodes that are still in the same group as the source or already in the same group as the destination. We can see that more than 50% of the nodes on the first hops (from the S-Group plot) are still in the source group of the message and only around 5% of the first hop nodes (from the D-Group plot) are in the same group as the destination. This explains why LABEL is not effective, since it is far from discovering the shortest path.

In order to further justify the significance of social based for-

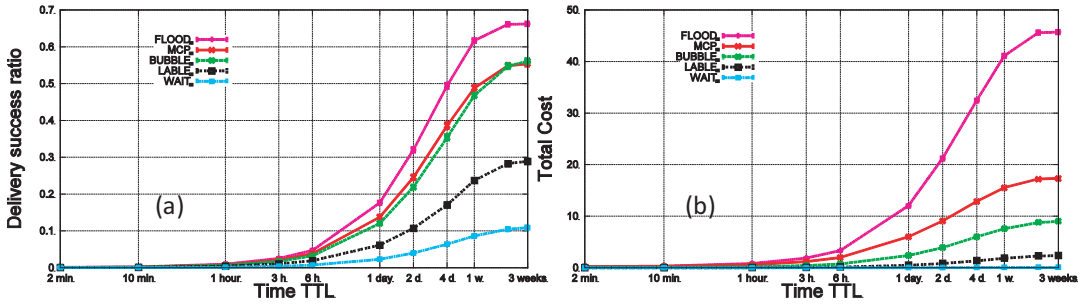

Figure 9: Comparisons of several algorithms on *Reality* dataset, all groups

warding, we also compare BUBBLE with a benchmark 'non-oblivious' forwarding algorithm, PROPHET[19]. PROPHET uses the history of encounters and transitivity to calculate the probability that a node can deliver a message to a particular destination. Since it has been evaluated against other algorithms before and has the same contact-based nature as BUBBLE (i.e. do not need location information), it is a good target to compare with BUBBLE.

PROPHET has four parameters. We use the default PROPHET parameters as recommended in [19]. However, one parameter that should be noted is the time elapsed unit used to age the contact probabilities. The appropriate time unit used differs depending on the application and the expected delays in the network. Here, we age the contact probabilities at every new contact. In a real application, this would be a more practical approach since we do not want to continuously run a thread to monitor each node entry in the table and age them separately at different time.

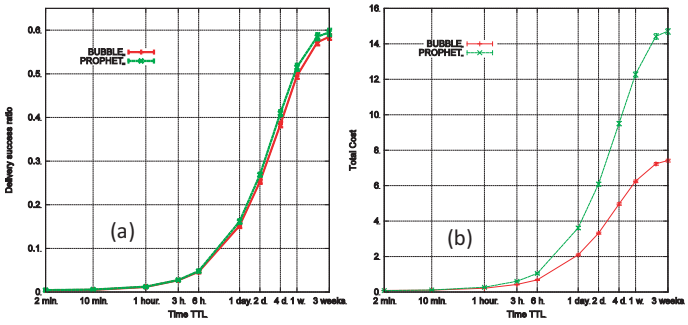

Figure 11: Comparisons of BUBBLE and PROPHET on *Reality* dataset

Figure 11 (a) and (b) shows the comparison of the delivery ratio and delivery cost of BUBBLE and PROPHET. Here, for the delivery cost, we only count the number of copies created in the system for each message as we have done before for the comparison with the 'oblivious' algorithms. We did not count the control traffic created by PROPHET for exchanging routing table during each encounter, which can be huge if the system is large (PROPHET uses flat addressing for each node and its routing table contains entry for each known node). We can see that most of the time, BUBBLE achieves a similar delivery ratio to PROPHET, but with only half of the cost.

Considering that BUBBLE does not need to keep and update an routing table for each node pairs, the improvement is significant. Similar significant improvements by using BUBBLE are also observed in other datasets, these demonstrate the generality of the BUBBLE algorithm, but because of page limit, we can not include the results here.

6.3 Approximating Centrality

For convenience of notation and evaluation of distributed centrality, we introduce an algorithm called RANK here, which is a component of BUBBLE, using only centrality information. In RANK, messages are pushed to nodes which have a higher ranking than the current node, until either they reach the destinations or they expire. In order to verify that the DEGREE is as good as or close to the centralise centrality, we ran another set of emulations using DEGREE instead of the pre-calculated centrality (i.e.RANK). We find out that RANK and DEGREE perform almost the same with the delivery and cost lines overlapping each other. They not only have similar delivery ratios but also similar costs.

For S-Window and C-Window, we can see from Figure 12(a) and (b) that the S-Window approach reflects more recent context and achieves maximum of 4% improvement in delivery ratio than RANK, but at double the cost. The C-Window approach measures more of the cumulative effect, and gives more stable statistics about the average activeness of a node. However, its cumulative measurement is not as good an estimate as RANK, which averages throughout the whole experimental period. It does not achieve as good delivery as RANK (not more than 10% less in term of delivery), but it also has lower cost. C-Window is easy to implement in reality and has similar delivery and cost to RANK (pre-calculated centrality), which is why we chose it for DiBuBB in Section 5.

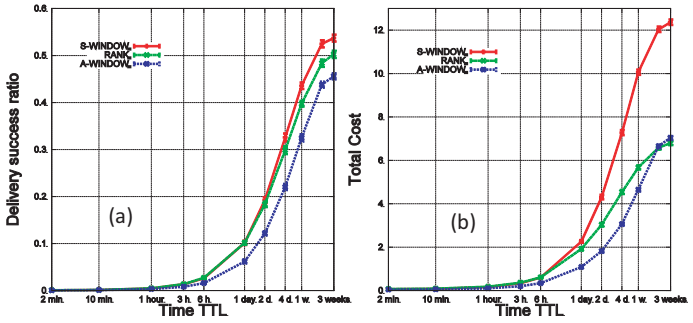

Figure 12: Comparisons of delivery (left) and cost (right) of RANK, S-Window and C-Window (*Reality*)

Furthermore, we want to verify whether the centrality measured in the past is useful as a predictor for the future. As well as the subset of the data we used in section 6.2, we extracted another two 3-week sessions from the dataset. We run a set of greedy RANK emulations on these, but using the centrality values from section 6.2. We found out that the delivery ratio and cost of RANK on the 2nd data session is as good as in the original dataset. Similar performance is also observed in the 3rd data session. These results imply some level of human mobility predictability, and show empirically

that past contact information can be used in the future. This past centrality can be used as a compliment of C-Window for DiBuBB, regarding that C-Window is around 10% less in term of delivery than RANK.

7. RELATED WORK

Several efficient forwarding algorithms for DTNs have been proposed. A majority of the algorithms are based on epidemic routing protocols [30], where messages are simply flooded when a node encounters another node. The optimisation of epidemic routing by reducing the number of copies of the message has been explored. For example, in [29], spray and wait routing assigns a limited number of copies. Many approaches calculate the probability of delivery to the destination node, where the metrics are derived from the history of node contacts, spatial information and so forth. The pattern-based Mobyspace Routing by Leguay *et al.* [17], location-based routing by Lebrun *et al.* [16], context-based forwarding by Musolesi *et al.* [22] and PROPHET Routing [19] fall into this category. PROPHET uses past encounters to predict the probability of future encounters. The transitive nature of encounters is exploited, where indirectly encountering the destination node is evaluated. Message Ferry by Zhao *et al.* [33] takes a different approach by controlling the movement of each node.

Recent attempts to uncover a hidden stable network structure in DTNs such as social networks have been emerged. For example, SimBet Routing [3] uses ego-centric centrality and its social similarity. Messages are forwarded towards the node with higher centrality to increase the possibility of finding the potential carrier to the final destination. In [11], we use small labels to help forwarding in PSNs based on the simple intuition that people belonging to the same community are likely to meet frequently, and thus act as suitable forwarders for messages destined for members of the same community. The evaluation demonstrates that even such a basic approach results in a significant reduction in routing overheads. RANK algorithm introduced in this paper uses betweenness centrality in a similar manner to SimBet routing. On the other hand, BUBBLE exploits further community structures and combines it with RANK for further improvement of forwarding algorithms. We have also exploit the closeness centrality to build an overlay over communities for multi-point asynchronous communications [32]. The mobility-assisted Island Hopping forwarding [23] uses network partitions that arise due to the distribution of nodes in space. Their clustering approach is based on the significant locations for the nodes and not for clustering nodes themselves. Clustering nodes is a complex task to understand the network structure for aid of forwarding.

In this paper, we have also shown how to uncover social structures from real world human connectivity traces. Discovering cliques or tightly connected clusters, i.e. communities, by looking for similar relation has also been studied in social network research such as World Wide Web [7], biological networks [9], social networks [25], and the Internet [20]. Graphs are a powerful tool to represent social relations and are structured in a quantified and measurable manner. The recent reviews [25] and [4] serve as introductory reading in community detection methods. Our current approach uses the duration and frequency of node connection for community definition and exploring further discovery of social community structure including further social contexts is left as future work. Finally, we emphasise that we take an experimental rather than theoretical approach, which makes a further difference from the other work described above.

8. CONCLUSION AND FUTURE WORK

We have shown that it is possible to detect characteristic properties of social grouping in a decentralised fashion from a diverse set of real world traces. We have demonstrated that such characteristics can be effectively used in forwarding decisions. Our algorithms are designed for a delay-tolerant network environment, built out of human-carried devices, and we have shown that they have similar delivery ratio to, but much lower resource utilisation than flooding, control flooding, and PROPHET.

Our decentralised approximation for centrality relates to the predictability of human mobility. We have made an additional contribution in this area by using similarity measures such as the Jaccard index [13]. Further improvement would entail more work on theoretical aspects of graph similarity [1]. In Section 5 we chose 6 hours as basic unit of centrality approximation. This appears to work well on the datasets we used; however, in future work we will examine the sensitivity of the system to this choice of period.

On forwarding, we have not directly emulated the distributed BUBBLE in this paper. Instead, we chose a divide-and-conquer method, showing separately the feasibility of decentralised approximation of centrality due to its inherent predictability. In principle, BUBBLE is supposed to work with a hierarchical community structure, but because of the limited size of data (each experiment is not large enough for us to extract hierarchical structure), the current algorithm and evaluation focus on a flat community structure. This can later be extended to a hierarchical structure. We will further verify our results when more mobility traces are available.

We believe that this paper represents a first step in combining rich multi-level information of social structures and interactions to drive novel and effective means for disseminating data in DTNs. A great deal of future research can follow.

9. ACKNOWLEDGEMENT

This research is funded in part by the ITA project and the Haggle project under the EU grant IST-4-027918. We would like also to acknowledge comments from Steven Hand, Brad Karp, Frank Kelly, Richard Mortier, Pietro Lio, Andrew Moore, Nishanth Sastry, Derek Murray, Sid Chau, Andrea Passarella, and Hamed Haddadi.

10. REFERENCES

[1] V. D. Blondel and P. V. Dooren. A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM Rev.*, 46(4):647–666, July 2004.

[2] A. Chaintreau, P. Hui, et al. Impact of human mobility on the design of opportunistic forwarding algorithms. In *Proc. INFOCOM*, April 2006.

[3] E. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proceedings of ACM MobiHoc*, 2007.

[4] L. Danon, J. Duch, et al. Comparing community structure identification. *J. Stat. Mech.*, page P09008, Oct 2005.

[5] N. Eagle and A. Pentland. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, V10(4):255–268, May 2006.

[6] K. Fall. A delay-tolerant network architecture for challenged internets. In *Proc. SIGCOMM*, 2003.

[7] G. W. Flake, S. Lawrence, et al. Self-organization of the web and identification of communities. *IEEE Computer*, 35(3):66–71, 2002.

[8] L. C. Freeman. A set of measuring centrality based on betweenness. *Sociometry*, 40:35–41, 1977.

[9] L. H. Hartwell, J. J. Hopfield, et al. From molecular to modular cell biology. *Nature*, 402(6761 Suppl), December 1999.

[10] P. Hui, A. Chaintreau, et al. Pocket switched networks and human mobility in conference environments. In *Proc. WDTN*, 2005.

[11] P. Hui and J. Crowcroft. How small labels create big improvements. In *Proc. IEEE ICMAN*, March 2007.

[12] P. Hui, E. Yoneki, et al. Distributed community detection in delay tolerant networks. In *Sigcomm Workshop MobiArch '07*, August 2007.

[13] P. Jaccard. *Bulletin de la Societe Vaudoise des Sciences Naturelles*, 37:547, 1901.

[14] E. P. C. Jones, L. Li, and P. A. S. Ward. Practical routing in delay-tolerant networks. In *Proc. WDTN*, 2005.

[15] D. Kempe et al. Connectivity and inference problems for temporal networks. *J. Comput. Syst. Sci.*, 64(4):820–842, 2002.

[16] J. Lebrun, C.-N. Chuah, et al. Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks. *IEEE VTC*, 4:2289–2293, 2005.

[17] J. Leguay, T. Friedman, et al. Evaluating mobility pattern space routing for DTNs. In *Proc. INFOCOM*, 2006.

[18] J. Leguay, A. Lindgren, et al. Opportunistic content distribution in an urban setting. In *ACM CHANTS*, pages 205–212, 2006.

[19] A. Lindgren, A. Doria, et al. Probabilistic routing in intermittently connected networks. In *Proc. SAPIR*, 2004.

[20] D. Lusseau and M. E. J. Newman. Identifying the role that individual animals play in their social network. *PROC.R.SOC.LONDON B*, 271:S477, 2004.

[21] A. Mislove, K. P. Gummadi, and P. Druschel. Exploiting social networks for internet search. In *Proceedings of the 5th Workshop on Hot Topics in Networks (HotNets'06)*, November 2006.

[22] M. Musolesi, S. Hailes, et al. Adaptive routing for intermittently connected mobile ad hoc networks. In *Proc. WOWMOM*, 2005.

[23] M. P. N. Sarafijanovic-Djukic and M. Grossglauser. Island hopping: Efficient mobility-assisted forwarding in partitioned networks. In *IEEE SECON*, 2006.

[24] M. E. J. Newman. Analysis of weighted networks. *Physical Review E*, 70:056131, 2004.

[25] M. E. J. Newman. Detecting community structure in networks. *Eur. Phys. J. B*, 38:321–330, 2004.

[26] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69, February 2004.

[27] S. Okasha. Altruism, group selection and correlated interaction. *British Journal for the Philosophy of Science*, 56(4):703–725, December 2005.

[28] G. Palla, I. Derenyi, et al. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[29] T. Spyropoulos, K. Psounis, et al. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *Proc. WDTN*, 2005.

[30] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke University, April 2000.

[31] P. Winters. Forecasting sales by exponentially weighted moving averages. *Management Science*, 6:324–342, 1960.

[32] E. Yoneki, P. Hui, S. Chan, and J. Crowcroft. A socio-aware overlay for multi-point asynchronous communication in delay tolerant networks. In *Proc. MSWiM*, 2007.

[33] W. Zhao, M. Ammar, et al. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proc. of ACM MOBIHOC*, 2004.