A Socio-Aware Overlay for Publish/Subscribe Communication in Delay Tolerant Networks

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

ABSTRACT

The emergence of Delay Tolerant Networks (DTNs) has culminated in a new generation of wireless networking. We focus on a type of human-to-human communication in DTNs, where human behaviour exhibits the characteristics of networks by forming a community. We show the characteristics of such networks from extensive study of real-world human connectivity traces. We exploit distributed community detection from the trace and propose a *Socio-Aware Overlay* over detected communities for publish/subscribe communication. Centrality nodes have the best visibility to the other nodes in the network. We create an overlay with such centrality nodes from communities. Distributed community detection operates when nodes (i.e. devices) are in contact by gossipping, and subscription propagation is performed along with this operation. We validate our message dissemination algorithms for publish/subscribe with connectivity traces.

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

Pervasive computing, Delay Tolerant Networks, Connectivity Modelling and Analysis, Network Measurement, Social Networks

1. INTRODUCTION

Wireless networking has moved from a first generation of wireless access provided by 802.11 LANs and cellular services, and a second generation of Mobile Ad Hoc Networks (MANETs), to a third generation: Pocket Switched Networks (PSNs) [5] are a category of Delay Tolerant Networks (DTNs) [21] aimed at supporting applications for human-to-human communication. Portable devices (e.g. smart phone, PDA) will be carried by most people in the future and communication is becoming more pervasive and autonomous in an opportunistic manner. In such environments, mobile nodes (i.e. devices) are sparsely distributed and networks are often partitioned due to geographical separation or node movement. DTNs employ a store-and-forward mechanism and opportunistic message dissemination to support network partitioning situations.

The goal of this paper is to introduce a novel approach for constructing a backbone for publish/subscribe communication based on uncovered human community structure in pervasive computing. We attempt to enable simple but powerful multi-point asynchronous communication in DTNs. In order to make DTNs viable, understanding both human interaction and mobility is necessary, to give us information for designing feasible applications and communication algorithms. Thus, we look into real-world human connectivity traces extracting characteristics of node interaction that lead to detection of communities from the traces. Several research projects such as the European Haggle project [11] and the MIT Reality Mining project [12] have collected contact based human connectivity traces using Bluetooth iMotes and cellular phones. These traces capture the human contact patterns over a wide range of periods and hence are useful for human interaction studies.

A key difference between traditional networks and DTNs is that an end-to-end path is expected to exist in traditional networks within a communication range, while DTNs allow looser connections between source and destination. Network storage allows DTN nodes to buffer data until connections are available. Thus, a node carries data until it encounters a node to pass it.

DTN research shares a similar paradigm with asynchronous messaging in middleware research. When nodes in a DTN have a local or global connection opportunity, messages are forwarded according to some policy, with the intention that they are brought 'closer' to their destination. Several trials have been performed for DTN forwarding algorithms, from simple flooding to using social networks. Because the environments where DTNs are deployed will be well integrated in daily life (e.g. VANET, Smart Phone CO_2 monitoring, disease epidemic spread monitoring), it is important to adopt a people-centric approach to model network semantics. Potential users include humans, vehicles, buses, trains and software agents, which must be part of a communication mechanism. They inject various types of data (i.e. context, event) for routing decisions. Thus, the network will be more human and data-centric, and will be integrated in various aspects, including reflection of human behaviour (mobility, membership, trust, etc.), visualisation with interaction, profiling (past history) and traceability.

The current research in DTNs focuses on the end-to-end communication, but many-to-many, any-to-many and one-to-any communication paradigms must be addressed, because typical communication in DTNs may be more group oriented. Multi-point communication (e.g. publish/subscribe) will provide aids to applications over DTNs such as smart caching. Smart caching is essential to provide prompt information availability and can be built based on social networks, where communities establish the backbone for content sharing in disconnected environments (e.g. Ad Hoc Google). We envision that the future communication structure in pervasive computing will be built in an incremental manner from small communities to a large urban communication space rather than devices follow the infrastructure-based networking.

This paper's contribution is twofold. First, we perform analysis of connectivity traces to uncover the characteristics of the networks including extracting communities and their centralities. Community detection is implemented using gossipping when nodes are in contact in a distributed fashion. Second, we propose a *Socio-Aware Overlay* for a message broker network using the centrality of a community, which is derived from distributed community detection. The overlay is a backbone for publish/subscribe communication and experiments with the connectivity traces are reported.

This paper continues as follows: Section 2 briefly describes the background and related works. Further background discussion can be found in the corresponding sections. Section 3 describes the network analysis of connectivity traces, Section 4 describes algorithms of community detection, and Section 5 introduces our *Socio-Aware Overlay* for publish/subscribe communication system. Section 6 contains conclusions and future work.

2. BACKGROUND AND RELATED WORK

In DTNs, the exact contact information between any nodes cannot be known in advance, and the routing decision at a node is difficult. In [39], a good summary of routing strategies in DTNs is given. Several strategies apply some degree of computation to deal with complexity of network semantics such as location tracking or mobility. There have been several social based forwarding studies in DTNs (e.g. [26]). However, there are not yet message forwarding algorithms derived from in-depth understanding of social networks, which is important because in PSNs a network node essentially represents a human. Most social network based forwarding algorithms simply follow the mathematical model constructed in a simulated environment. Thus, more study of social network aspects in real world scenarios is desired. We take an empirical approach and work directly with the real world connectivity traces by analysis and validation.

Multi-point asynchronous communication is useful for many distributed applications over DTNs such as resource discovery, where destination of communication can be one or many in a group, and asynchronous operation is preferable. Communication types include many-to-many (e.g. publish/subscribe), any-to-many (e.g. multicasting) and one-to-any (e.g. anycasting). Publish/subscribe is a powerful abstraction for building distributed applications. Communication is message-based and can be anonymous, where participants are decoupled from the following aspects:

- Space - no direct connection between clients
- Flow - no synchronised operation is required on event publishing and subscribing
- Time - no need to be running at the same time

Thus, decoupling characteristics give the advantage of removal of static dependencies in a distributed environment. It is a good solution to support highly dynamic, decentralised systems. Most distributed event-based middleware supporting a publish/subscribe paradigm contains three main elements: a publisher who publishes events (messages), a subscriber who subscribes his interests to the system, and an event broker network to match and deliver the events to the corresponding subscribers. Event brokers are usually connected in an arbitrary topology. In a distributed event-based middleware, the event brokers form an agent network providing routing, event matching, and filtering services [3] (see Fig. 1).

Publish/subscribe shares similar issues with MANET multicast [24]. The basic idea to define multicast routing in MANETs is to form a path to all group members with minimal redundancy. It is also critical whether the routing table is constructed on-demand, or optimal paths are determined once and updated periodically. Control packets can be flooded throughout the network or limited to the nodes in the multicast delivery tree.

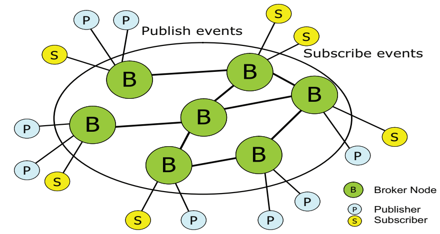


Figure 1: Distributed Publish/Subscribe System

Dynamic wireless network environments seem to require more dynamic multicast group creation based on the message contents instead of a pre-assigned channel. As a result, the groups tend to be smaller, frequently short-lived, and more numerous. This is significantly different from group membership in traditional multicast, where groups are defined in advance and only the membership is dynamic (see [37] for multicast membership in MANETs).

Potential applications using publish/subscribe vary including environmental monitoring by pervasive devices. The MetroSense project [9] explores the use of people-centric sensing with personal as well as consumer oriented sensing applications. Sensing can potentially cover a campus, city, or a whole metropolitan area, with many potential applications such as noise mapping and pollution mapping. Similarly, the urban sensing project CENS [33] seeks to develop cultural and technological approaches for using embedded and mobile sensing to invigorate public space and enhance civic life.

A social network consists of a set of people forming social meaningful relationships, where prominent patterns or information flow are observed. In PSNs, social networks could map to computer networks since people carry the computer devices. Many experiments captured this type of network connectivity trace, which is shown in the next section. Understanding the whole network characteristics is a popular study. Discovering cliques or tightly connected clusters by looking for similar relations are also common studies in social network research [2][35]. Graphs are a powerful tool to represent social relations and are structured in quantified and measurable manner.

3. NETWORK ANALYSIS

A key requirement for human interaction in pervasive environments is capturing trace data from the real world (e.g. human connectivity and intermittency of connections between people) in order to construct realistic synthetic models. For example, the Reality Mining project collected proximity, location and activity information, with nearby nodes being discovered through periodic Bluetooth scans and location information from cell tower IDs. Several other groups have performed similar studies. Most of these, such as [12], [11] and [28], use Bluetooth to measure device connectivity, while others, such as [15], rely on WiFi. The duration of experiments varies from 2 days to over 300 days, and the numbers of participants vary from 8 to over 5000. The Crawdad database [10] provides extensive traces, which are useful for the validation of forwarding algorithms and routing protocols that operate through learning characteristics of node mobility. Some traces include location information (e.g. MIT), however the majority of traces have only node connectivity information (e.g. Haggle). Thus, if location information is available, it is possible to infer the mobility of nodes. In this paper, we focus on connectivity in the traces, and leave investigation of geographical information as future work.

Note that it is a complex task to collect accurate connectivity traces using Bluetooth communication, as the device discovery protocol may limit detection of all the devices nearby. Bluetooth inquiry can only happen in 1.28 second intervals. 4×1.28 (i.e. 5.12 seconds) gives you more than 90% chance of finding a device. However there is no data available when there are many devices and many human bodies around. Power consumption of Bluetooth also limits scanning interval if devices have limited recharging capability. iMote connectivity

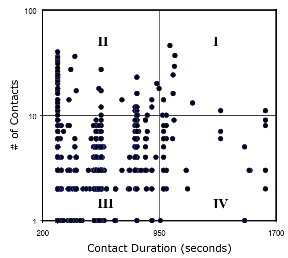
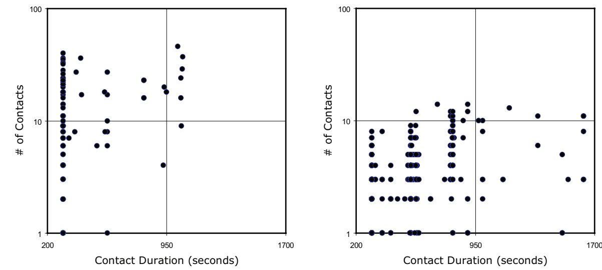


Figure 2: Node Classification (Wireless Rope)



a. Tracking Station **b. Personal Device**

Figure 3: Node Characteristics

traces in Haggle use around 2 minutes scanning interval, while the Reality Mining project uses 5 minutes. See [29] for further details related to connectivity data collection with Bluetooth.

Previously the characteristics of these data, such as inter-contact and contact distribution, have been explored in several studies [4] [18] [25], to which we refer the reader for further background information. In this paper, we focus on extracting information related to levels of clustering or network transitivity and strong community structure. We have analysed various traces from Crawdad database and show analysis with the MIT Reality Mining Project [12], the UCSD wireless topology discovery project [34], the Haggle project [11], and conference activity data, by Wireless Rope [28]. Note that connectivity trace data may not be perfect for the moment when mobile phones are used to gather anonymous devices since only an average 15% of population turn on the Bluetooth communication. A brief explanation of the trace data is given below:

MIT in the MIT Reality Mining project [12], 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, as well as logging information about the cellular tower they are associated with.

UCSD in the UCSD Wireless Topology Discovery [34], approximately 300 wireless PDAs running Windows Pocket PC were used collecting WiFi access points information periodically for 11 weeks.

CAM in the Cambridge Haggle project, 40 iMotes were deployed to 1st year and 2nd year undergraduate students for 11 days. iMote detects proximity by Bluetooth.

WirelessRope Wireless Rope [28] is a tool to detect social situations by Bluetooth proximity with consumer devices and its effects on group dynamics. The logged data comprises Bluetooth name, Service class, sighting information, IDs, and original/transformed time.

The connectivity traces can be represented in the form of weighted graphs called contact graphs, with the weight of an edge representing the contact duration/contact frequency for the two end vertices. Hence understanding human interaction can be tackled from the domain of weighted network analysis. Possible outcomes from studying of the weighted contact graphs include community detection and weighted node centrality. Many real-life networks are weighted, but because of complexity, little analysis has been done in this area. The seminal work is a weighted network analysis paper by Newman [22]. A weighted graph can be converted into a multi-graph with many unit edges. One can then apply the usual non-weighted versions of various algorithms, including a community detection algorithm based on edge betweenness (for more detail see Section 4).

3.1 Inter-Contact Time

For a given pair of nodes A and B, the time-line can be divided into two regions, contact times and inter-contact times. The contact times are when A and B are in range of one another, and could therefore have sent data if they had wished to. Inter-contact times are the times between the contact times, and the distribution of inter-contact times simply indicates the frequency of interactions. Our previous works [4] [19] have shown that inter-contact time follows a power-law distribution, where the higher that value of the power coefficient, the more frequently the node pairs interact.

In [18], we have shown that an intra-community pair has higher power law coefficient than an inter-community pair; that is, nodes pair in the same community tend to meet more often. See [4] and [19] for more details of inter-contact time analysis on the connectivity trace.

3.2 Four Pair Categories

The correlation between contact duration and the number of contacts can be split into the following four categories. Meetings take place between pairs of individuals at a rate which is high if a pair has one or more mutual friends, and low otherwise. Acquaintances between pairs of individuals who rarely meet decay over time. There is an upper limit on the number of friendships an individual can maintain. Proximity determines community in many cases; however, how to evaluate proximity or common interests is an issue still to be determined. In Wireless Rope, each person can define the criteria of four categories. In general, a more in-depth analysis in the following social contexts may reveal new aspects to consider. Fig. 2 depicts the four categories on a Wireless Rope trace.

I Community High number of contacts and longer contact duration

II Familiar Stranger High number of contacts and short contact duration

III Stranger Low number of contacts and short contact duration

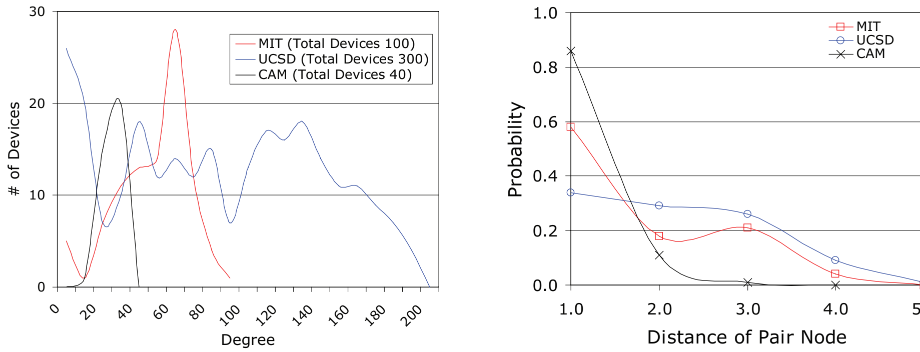
IV Friend Low number of contacts and longer contact duration

Nodes with *High Visibility* and *No Mobility* (e.g. Tracking Stations) will be good candidates for rendezvous nodes. Fig. 3 show the device characteristics. A tracking station has many familiar strangers but no friends, while the personal device shows a clear friend zone. This classification of nodes is the base for our current community detection by setting the number of contacts and duration as threshold values.

3.3 Node Centrality

Understanding a network and a node's participation in the network is important. For example, it is key to identify important actors in a social network, where actors are usually located in strategic locations within the network and have power to impact on others. These actors, or *centralities*, can be found out by measuring the network is essential. This gives insight into the roles and tasks of nodes in a network. Three well known centrality measures are: *Degree, Betweenness, and Closeness* Centrality.

Degree centrality measures the number of direct connections. This indicates that the node must be the most active in the network. Fig. 4a shows the degree distribution of MIT, UCSD, CAM traces. A high number of connections indicates that the node may be a good candidate to be a hub. MIT and CAM show strong scale-free network characteristics, where only certain nodes have high degrees.



a. Degree Centrality **b. Distance of Pair Nodes**

Figure 4: Degree and Distance of Pair Nodes

Betweenness centrality indicates a bridge node between two nonadjacent nodes. Thus, a high betweenness potentially might have control over these two nonadjacent nodes. A betweenness node in the network may impact on data flow between two communities. Use of betweenness centrality between communities is planned, but is out of scope of this paper.

Closeness centrality yields the node with the shortest path to all others and the best visibility in the network and sub-network (i.e. community). It is a measurement of how long it will take data to spread the others in the community. The closeness $C_C(a)$ for a vertex a is inverse sum of distances to other nodes b:

$$C_C(a) = \ 1 \ / \ \sum_b d_{ab} \tag{1}$$

Our *Socio-Aware Overlay* currently uses closeness centrality nodes as messaging brokers (see Section 5 for more details) so that the chosen broker maintains a higher message delivery rate. Understanding the characteristics of centrality is a basis for detecting community structure described in Section 4.

3.4 Distance Distribution

Fig. 4b depicts the number of hops between each pair of nodes in the MIT, UCSD, and CAM traces. Table 1 summarises average hop counts and cluster coefficient values of MIT, UCSD, and CAM traces. The cluster coefficient value of the MIT trace 0.44 indicates 44% chance that if node A knows nodes B and C, then nodes B and C know each other. We also examined the CityWare data [30], where proximity data is collected in city scale, and it shows an average hop count of 3.3 and cluster coefficient value of 0.45. These values indicate that the network structure is scale-free, which gives great promise for our proposed *Socio-Aware Overlay* approach.

Experimental traces	Average Hop Count	Cluster Coefficient
MIT	1.6	0.44
UCSD	2.2	0.41
CAM	1.2	0.66

Table 1: Average Hops and Cluster Coefficient

4. COMMUNITY DETECTION

People tend to form groups inherently in the structure of society and such groups evolve over time. We aim to uncover the structure and dynamics of such social communities from the human connectivity traces, where social groups must be embedded. There have been studies of identifying communities in physical environments, where community detection is based on the location or some context defining the community. On the other hand, a social community may not be visible in such physical environments, where people communicate by email or social network services. Dealing with human connectivity traces requires understanding both physical and virtual communities in such pervasive environments. We have shown various community detection mechanisms from human connectivity traces mostly in a centralised manner [17]. In this section, we show community detection in decentralised fashion, which becomes an important input for constructing an overlay for publish/subscribe systems.

Members of a community share the same interest with high probability; and understanding the community provides efficient routing and forwarding mechanisms. Detected communities from the traces may be static social communities or temporal communities such as a group of people who happen to be at the same conference. Our current approach does not distinguish between these two different community concepts and further consideration of community concepts along with membership management is part of our ongoing work. However, both types of detected communities contain an influential attribute for forwarding efficiency.

Community detection in complex networks has attracted a lot of attention in recent years. In biological networks, it is widely believed that the modular structure results from evolutionary constraints and plays a crucial role in biological functions [14]. In social networks, community structures corresponding to human social communities [27]. In the Internet, the community structures correspond to autonomous systems, which are connected segments of a network comprising a collection of subnetworks interconnected by a set of routers. In the DTNs, community structure corresponds to some human communities. Given the relevance of the problem, it is crucial to construct efficient procedures and algorithms for the identification of the community structure in a generic network. See the reviews by Newman [27] and Danon *et al.* [8] for methodological overviews and comparative study of different algorithms.

4.1 Distributed Community Detection

In a realistic DTN scenario, the existence of a centralised server to process the data can not be assumed. Thus, each node needs to detect its own local community. In [6], Clauset defines a measure of local community structure and an algorithm that infers the hierarchy of communities that encloses a given vertex by exploring the graph one vertex at a time. For graphs where exploring a new vertex is time-consuming, like the encounter pattern in DTNs, the running time is linear, *O(k)*, where *k* is the number of vertices in the local community.

In this section, we introduce two of our distributed community detection algorithms, named SIMPLE and k-CLIQUE. SIMPLE is our novel algorithm, which classifies nodes based on the number of contacts and contact duration of a node pair according to an *a priori* threshold value T_{th}. k-CLIQUE is based on [31] in which a k-clique community is defined as a union of all k-cliques (complete sub-graphs 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.

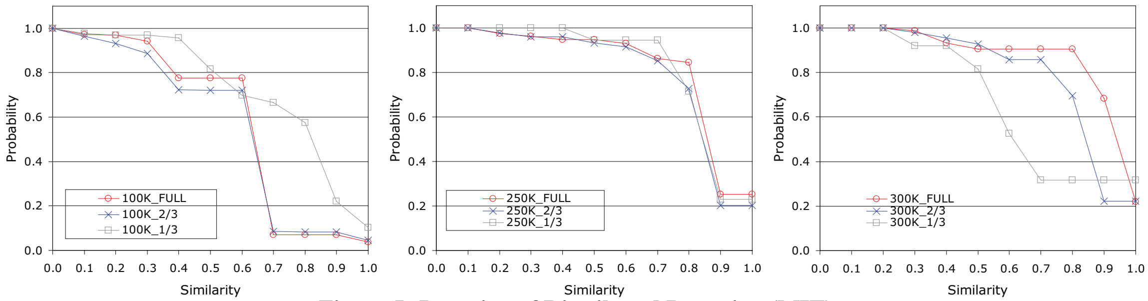


Figure 5: Duration of Distributed Detection (MIT)

4.2 Definitions

The common terminologies for our detection algorithms are:

Familiar set: we assume each vertex (mobile device) will keep a map of vertices it has encountered with the corresponding cumulative contact durations. When the cumulative contact duration with a vertex exceeds a certain threshold T_{th}, it is promoted to be included into its *familiar set* F. For a given vertex, v_i, perfect knowledge of its own familiar set is denoted F_{v_i} and incomplete knowledge of other vertices' familiar sets (e.g. a local approximation of the familiar set for vertex v_j) is denoted $\tilde{F}_{v_j}$.

Local Community: The local community of vertex F_{v_i}, denoted by C_{v_i}, contains all the vertices in its *familiar set* (its direct neighbours) and also the vertices that are selected by our following community detection algorithms.

The basic structure of our algorithms is as follows. When a mobile device v_0 first initialises its community detection procedure, the local community C_{v_0} only contains this source vertex. Whenever it encounters another device v_i, they will exchange part of their local knowledge of the network. v_0 then has to decide on the following based on certain acceptance criteria:

1. whether to place the encountered vertex v_i in its *familiar set* F_{v_0} and/or C_{v_0}.
2. whether C_{v_0} should merge with the whole or part of C_{v_i}.

Both algorithms we introduce here differ only in the admission criteria into the familiar set and local community.

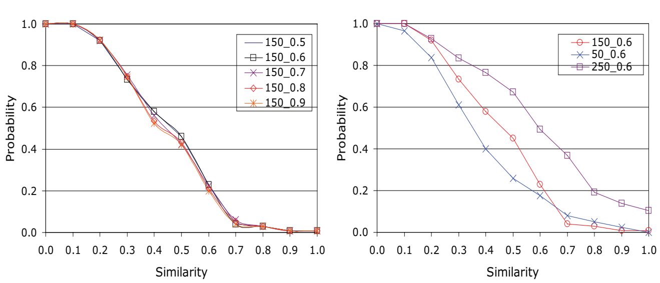
4.3 Algorithms

When a mobile device v_0 encounters another device v_i, the following algorithm will execute:

1. Each vertex, v_0, needs to maintain the following information: a list of encountered nodes and their contact durations (practically encounters that do not meet certain criteria will be discarded from the list), its *familiar set* F_{v_0} (its familiar set of vertices), its local community C_{v_0} detected so far, and

   (k-**CLIQUE**) a local approximation of the familiar sets of all vertices in its local community C_{v_0}:

   $$FoC(v_0) = \{\tilde{F}_{v_j} \mid v_j \in C_{v_0}\}$$



a. Ratio of Common Members **b. Contact Duration**

Figure 6: Impact of Different Threshold Criteria

2. *Initialisation:* $C_{v_0} \leftarrow \{v_0\}$, $F_{v_0} \leftarrow \emptyset$ and $FoC(v_0) \leftarrow \emptyset$

3. When v_0 encounters another v_i, they exchange local information, i.e. v_0 will acquire from v_i the following: C_{v_i}, F_{v_i} and (k-**CLIQUE**) $FoC(v_i)$

   Each local approximation of familiar set in $FoC(v_0)$ is merged (by taking the set union) with the corresponding versions just obtained from $FoC(v_i)$. e.g.

   $$\forall k\ s.t.\ \exists \tilde{F}_{v_k} \in FoC(v_0),$$

   replace

   $$\tilde{F}_{v_k}\ in\ FoC(v_0)$$

   with

   $$(\tilde{F}_{v_k} \in FoC(v_0)) \cup (\tilde{F}_{v_k} \in FoC(v_i))$$

4. If v_i is not in F_{v_0}, v_0 updates the total contact duration counter of v_i which is stored at v_0, until v_i falls out of contact and meanwhile the algorithm forks and proceeds to the next step (5). When the total contact duration count has exceed a certain threshold (a design parameter), v_0 will insert v_i in F_{v_0} and C_{v_0}.

5. If v_i is not in C_{v_0}, then add v_i to C_{v_0} if it satisfies the following algorithm-specific criteria:

   (**SIMPLE**) if $|F_{v_i} \cap C_{v_0}|/|F_{v_i}| > \lambda$ (where λ is the merging threshold which we will vary in this paper to see the different of final communities detected).

   (k-**CLIQUE**) if the *familiar set*, F_{v_i} contains at least $k-1$ members of the local community, C_{v_0}, i.e. if

   $$|F_{v_i} \cap C_{v_0}| \geq k - 1$$

6. If v_i is added to C_{v_0} in the previous steps, the aggressive variants of the algorithm behave as follows:

   (**SIMPLE**) if the number of vertices overlapping C_{v_0} and C_{v_i}, (i.e. $|C_{v_0} \cap C_{v_i}|$), is greater than γ of $|C_{v_0} \cup C_{v_i}|$ (γ is the merging threshold as well which can be different from λ in step 5, but we will use the same value for both cases in this section), then merge (by taking the set union of) the two communities. i.e. the merging criterion is

   $$|C_{v_0} \cap C_{v_i}| > \gamma |C_{v_0} \cup C_{v_i}|$$

   (k-**CLIQUE**) if the *familiar set*, $\tilde{F}_{v_j}$ of a vertex v_j inside the local community of v_i contains at least $k-1$ members of the local community of v_0, v_j is added into the local community C_{v_0}, i.e. if

   $$|\tilde{F}_{v_j} \cap C_{v_0}| \geq k - 1$$

   If this criteria is satisfied, then $FoC(v_0)$ also needs to be updated to include $\tilde{F}_{v_j}$.

Clearly, the SIMPLE algorithm require less storage and less computation than the k-CLIQUE algorithm.

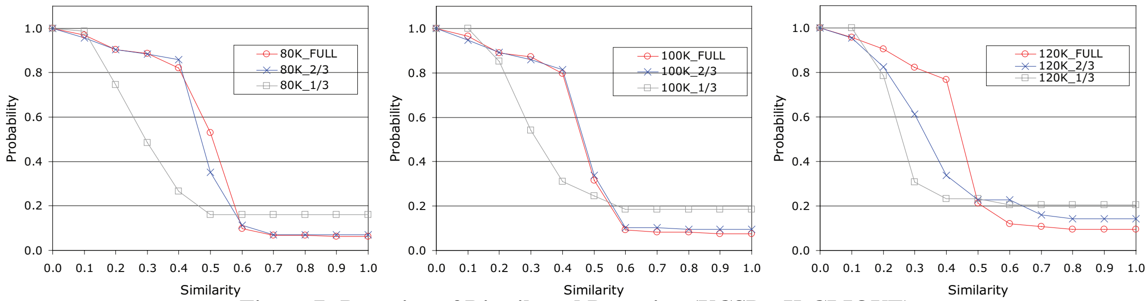


Figure 7: Duration of Distributed Detection (UCSD - K-CLIQUE)

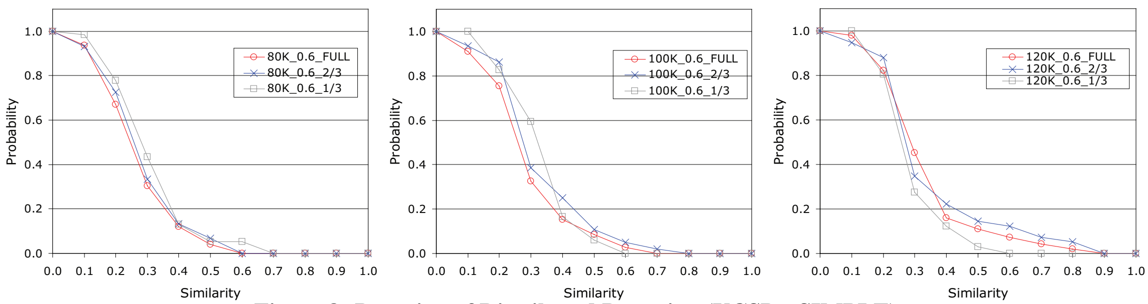


Figure 8: Duration of Distributed Detection (UCSD - SIMPLE)

Experimental traces	SIMPLE	k-CLIQUE	Communities
MIT	0.79/0.76	0.87	8
UCSD	0.47/0.56	0.55	8
CAM	0.85/0.85	0.85	2

Table 2: Summary of Distributed Community Detection

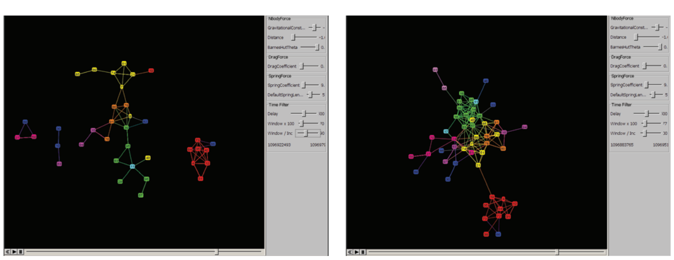
4.4 Evaluation

Table 2 summarises the highest similarity values calculated by each distributed algorithm. For SIMPLE, we show both its comparison with the centralised k-CLIQUE (first) and the centralised Newman method [22] (second). We can see the best performance of the algorithms can be up to 90% detection accuracy compared to the centralised methods. This gives a possibility that distributed community detection can be realised. k-CLIQUE has slightly better performance than its SIMPLE counterpart, because k-CLIQUE requires more information and calculation. Considering its computational and storage requirements, the performance of SIMPLE is acceptable. The complexity of SIMPLE is $O(n)$, and it may be suitable for resource constrained mobile devices. If the mobile devices can afford more storage, k-CLIQUE would be a good choice due to its reasonably good similarity values. We use modified version of the classic Jaccard index [32] for similarity measurement between two communities (see [20] for more detail). The *core communities* detected by distributed methods are compared with the communities detected by centralised algorithms using our similarity measurement.

Fig. 5-8 give some more detailed illustration of the results. Fig. 6 depicts the impact of different threshold criteria T_{th} on the accuracy of community detection using the k-CLIQUE algorithm. In Fig. 6a, the same value of contact duration is used (i.e. 150k seconds) with changing similarity level when two sets of communities are compared (i.e. the range 50% - 90%). It shows no significant change from this aspect. In Fig. 6b, on the other hand, the value of contact duration is changed (i.e. 50k - 250k seconds) and it indicates clearly that higher values cause significant improvement on the accuracy of community detection.

Fig. 5 depicts the same experiment running for the full length of the trace, 1/3, and 2/3 on three different contact duration criteria. In the best case with the 250K contact duration threshold, processing 1/3 of the trace shows a comparable result to the centralised approach. Fig. 7 depicts detection by the k-CLIQUE algorithm with UCSD data and Fig. 8 depicts detection by the SIMPLE algorithm with UCSD data. Both results show the stability with running 2/3 of trace processing. Further details of the detection algorithms and results with various traces can be found in [20].

Furthermore a sliding time window for community detection can be set, where the threshold value for community detection is evaluated within the specific time window (i.e. duration). Because of the space restriction, this topic is out of scope of this paper. However, we show the visualisation of detected communities based on the time window size in Fig. 9, in which the smaller time window depicts communities with fewer members, but probably they are tighter-knit communities than the larger time window. See [38] for further details of visualisation of community detection.



a. Small Time Window **b. Large Time Window**

Figure 9: Community Detection with Different Time Window Size

5. MULTI-POINT COMMUNICATION

Creating an overlay for message dissemination has been a popular technique for multi-point communication. Below, we present a brief discussion of existing approaches along gossip based approaches. This discussion leads to our proposal: *Socio-Aware Overlay*.

Overlay Approach: In [7], an *Overlay Tree* creates a dissemination tree and maintains it in response to changes in the topology by reconfiguring routes traversed by events. In [16], a distributed protocol to construct an optimised publish/subscribe tree in ad hoc wireless networks is presented. Each publisher node becomes a root in a multicast tree. Applying flooding (or random walk) over the physical topology graph is one way to find routes to an object with a target key.

Another possibility is to create topology dependent identifiers for the nodes and to apply geographical routing techniques (e.g. GPSR [23]). An object is stored and replicated at nodes near to the node where the key is stored. These approaches lack an understanding of the actual network structure and do not take advantage of what routing strategies can gain. Our *Socio-Aware Overlay* puts importance on consideration of the real situation of the network semantics. Thus, once the appropriate network structure is found (e.g. scale free networks), it should show a significant advantage.

Gossip Approach: Maintaining a tree topology is challenging, as it requires high network traffic to detect and repair failed links. Thus, a structureless approach is desirable, where no global network-wide structure and no link breakage detection are required. This approach is resilient to network partition. The epidemic dissemination mechanism is a powerful form of peer-to-peer (P2P) cooperation. Gossipping is a simple routing protocol, where the retransmission probability function is a constant value. In [13], this algorithm is extended, where probability 1 is given for the first k hops. This stops gossipping when only a few neighbours are near the gossip root node.

Most gossipping approaches lack consideration of the multi-point communication aspect. The control flooding approach implicitly indicates that the diffusion process can be managed by subscriptions in [36]. We now introduce our novel approach for multi-point communication supporting PSN environments.

5.1 Socio-Aware Overlay

We propose multi-point event dissemination using an overlay constructed by closeness centrality nodes in communities and name this overlay structure *Socio-Aware Overlay*. It takes a clustering-based approach and membership of the group is dynamically detected through a community detection process rather than implicitly defined as the set of nodes within a certain area in geographical or physical casting. Cluster-based algorithms partition a wireless network into several disjoint and equally sized regions, and select a cluster head in each region to operate message exchange.

Detected community members are well connected, implying that socially they share the same interests with high probability. Thus, similar subscriptions may coexist within the same community. The fundamental idea of this approach is instead of artificially constructing an overlay based on various contexts (e.g. location, group mobility), the existing structure is detected and mapped to the function. Thus, this approach strongly depends on dynamic community detection, and a crucial factor is the quality of the community detection mechanism. Our current community detection algorithms detect approximately 80% of communities compared to the centralised approach.

At the same time, subscription propagation is operated during the community detection by gossipping when two nodes are in contact, which does not cause any extra cost. State maintenance requires control traffic, which could be expensive to operate, while a stateless approach could also be expensive if using event flooding. Stateful approaches suffer under frequent topology changes, whereas stateless approaches are more suitable for topology change and the partitioning and isolation of nodes. In a stateless approach, the gossip dissemination sends each message to a randomly chosen group of nodes. Thus, our approach takes advantage of both stateful and stateless approaches to deal with dynamic network environments.

Structured overlays assign identifiers to nodes, and control the identifiers of neighbours in overlay networks and the keys of the objects they store. This is effective since lookups can be done with cost $O(logN)$, which is better than a flooding approach. However, the characteristics of dynamic mobile networks require a significant amount of traffic to maintain the overlay links. In [1], a structured P2P overlay network is used for a publish/subscribe system. Subscriptions are mapped to keys and sent to a rendezvous node. The performance of this approach depends on the real mapping between the overlay network and the underlying network topology. Our *Socio-Aware Overlay* is mapped over detected communities, which gives a certain level of stable network topology (see Section 3 for network characteristics).

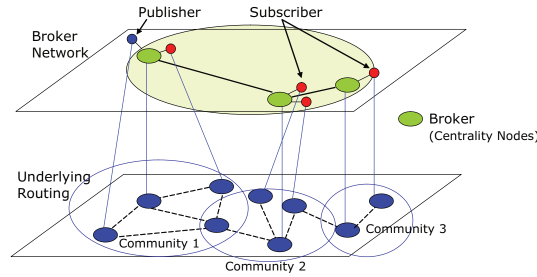


Figure 10: Overlay over Communities

We currently choose a closeness centrality node for the broker node as closeness centrality implies the best visibility in the community. Thus, once this node gets the message, delivery to any member of the community has high reliability. Because of the characteristics of human networks (i.e. scale-free networks), many nodes within a community are tightly connected and multiple closeness centrality nodes can coexist. This is an advantage as it potentially balances the workload of brokers and it will be the subject of future work to add a load balancing mechanism.

The proposed multi-point communication takes advantage of PSNs, where various communication methods can be used to control delay in PSNs. Communication between brokers can have two modes: *Unicast* and *Direct*. *Unicast* is based on the underlying unicast algorithms. Thus, it could end up as epidemic routing. *Direct* provides a more direct communication mechanism such as WiFi access points or GPRS. The *Direct* approach gives accelerated message delivery with some cost. When *Unicast* is used for the communication between broker nodes, the average hop count follows the distance of the pair nodes (i.e. 1.6 hops for MIT Reality mining trace). Using the betweenness centrality, where a node has dual visibility from and to communities, will improve the hop counts.

5.2 Overlay Construction

Fig. 10 depicts a publish/subscribe broker overlay, which is dynamically constructed through the gossipping stage for the community detection. Construction of the broker overlay is independent from underlying unicast routing algorithms. In the following, we briefly sketch the algorithm that realises the proposed communication mechanism.

1. Distributed community detection operates gossipping between nodes when in contact. The size of the exchanged data is small, and the contact duration is assumed to be enough to complete an exchange. Besides the community detection data the following information is exchanged.

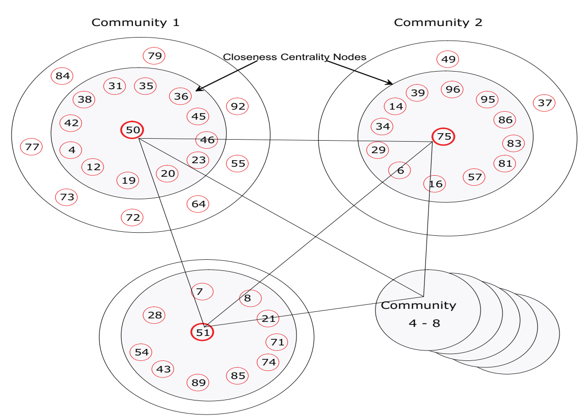


Figure 11: Community Structure

# Pub/Sub	Average Hops	Contact to Sub (B→C)	Pub to Sub (A→C)	Latency	Undelivered	Total Hops
1000/100	1.27	0.0 units	74.2 units	8.0H	174(17%)	6528
500/50	1.26	0.0 units	67.5 units	7.3H	107(21%)	1852
200/20	1.45	0.0 units	21.9 units	2.4H	81(41%)	351

Table 3: Event Dissemination with Socio-Aware Overlay (CAM)

# Pub/Sub	Average Hops	Contact to Sub (B→C)	Pub to Sub (A→C)	Latency	Undelivered	Total Hops
1000/100	1.28	5.6 units	631.6 units	68.4H	261(26%)	6431
500/50	1.34	4.6 units	828.5 units	89.8H	242(48%)	1373
200/20	1.32	4.3 units	831.4 units	90.1H	115(58%)	204
1000/100C	1.35	2.7 units	449.4 units	48.7H	33(3%)	-

Table 4: Event Dissemination with Socio-Aware Overlay (MIT)

# Pub/Sub	Average Hops	Contact to Sub (B→C)	Pub to Sub (A→C)	Latency	Undelivered	Total Hops
1000/100	1.01	0.0 units	645.7 units	70.0H	846(85%)	237
500/50	1.04	0.0 units	988.1 units	107.0H	432(86%)	85
200/20	1.00	0.0 units	1660.8 units	180.0H	183(92%)	17

Table 5: Event Dissemination with Socio-Aware Overlay (UCSD)

- Subscriptions/unsubscriptions with the destination of the respective community broker nodes.
- A list of centralities with timestamp.

2. Each node has a local view of the community and calculates closeness centrality as the corresponding message broker node.

3. When a broker node changes upon calculation of closeness centrality, the subscription list is transferred from the old one to the new one. A broker node information update is sent to all the brokers.

4. During gossipping, subscriptions are also propagated towards the closeness centrality node in the community.

   After operations 1 - 4:

   - Each node knows its broker node.
   - Each node keeps its own subscriptions.
   - Each broker node keeps the community's subscription list. It may keep the subscription list of individual subscribers or an aggregated subscription list.
   - All broker nodes keep the list of brokers in the other communities. The list is collected during gossipping.

5. Once the publication is given from the publisher node to the broker node, the broker node propagates it by one of two communication modes (i.e. *Unicast* or *Direct*) to all the brokers.

6. When a node has a publication, it sends it to its broker node within the community.

7. When the node without any community has a subscription or a publication, currently the default community is assigned.

8. When the broker node receives a publication, it operates matching against the subscription list. If it matches, it floods the publication within the community. This operation may be done either unicast or broadcast. The broker has knowledge of the average/max hops to all the members of the community. When the broker uses broadcast, max hops can be used to control flooding. The detail of this operation is out of scope of this paper.

Multiple centrality nodes can be used as a group of brokers. In the next section we show preliminary results of the publish/subscribe simulation to give an idea on whether our algorithms for publish/subscribe in DTNs are feasible. We have performed a series of community detection against different connectivity traces [20]. Detected numbers of communities are shown in Table 2. Fig. 11 depicts the community structure and closeness centralities detected in the MIT trace. Eight communities are detected, and the largest *Community 1* contains 21 members. *Communities 4-8* contain 3-4 members each. There are 24 devices that do not belong to any communities, named *Loners*. Multiple centrality nodes are selected, which are in the inner circle. In *Community 1*, 13 nodes are closeness centrality nodes, and therefore broker as a group. Alternatively, one of the centrality nodes is named as a single broker, which is marked at the centre of the circle in Fig. 11. All the detected centrality nodes have a single hop count to all members of the community, and an average of 93% of nodes in the community can take the role of event broker. Note that 24 nodes (25% of community members) do not belong to any communities.

Membership: Because of the delay of delivery, group membership to the topic may change during message propagation operation. Subscription information needs to be updated dynamically. The current membership model is based on subscription information propagated by a gossipping mechanism. Thus, in case when subscription is no longer valid for the subscriber but if the publisher has not obtained un-subscription information, the message is attempted to be delivered. There are two concepts for the membership to be considered: (i) *Membership* of the community, and (ii) *Membership* to shared contents: *Topics* for publish/subscribe. *Community* is a permanent persistent entity, *Topics* may often map to community, because members share the same interests. Thus, managing membership of the community indirectly controls part of publish/subscribe functionality. The *Socio-Aware Overlay* approach currently provides membership management in an implicit way by distributed community detection. Each local view of community change reflects a community membership change. However, expiration of membership, membership refresh, or change of roles within a community is not yet completely managed.

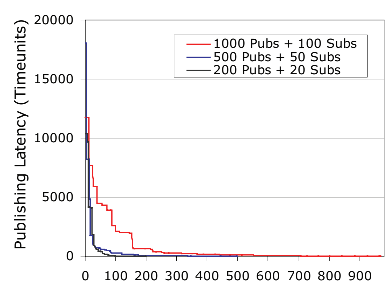


Figure 12: Latency of Publications I

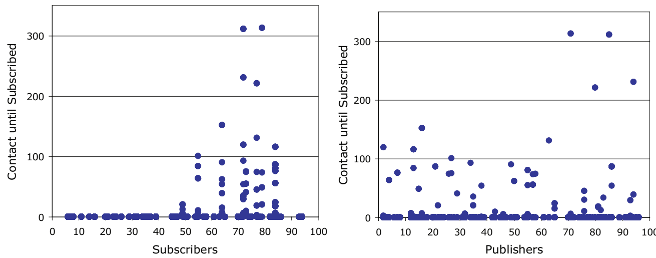
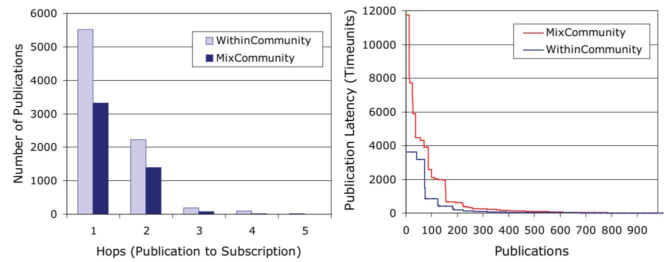


Figure 13: Latency of Publications II



a. Hop Counts/Publication **b. Publication Latency**

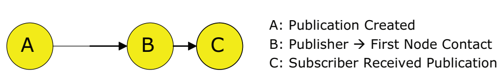
Figure 14: Latency: Within and Mix Community

5.3 Results and Discussion

For validation and evaluation of the proposed approach, we use a discrete event emulator to replay the connectivity traces. The original trace files are divided into discrete sequential contact events and fed into the emulator as inputs. Although the current subscription model is simply topic-based, content-based filtering can be operated in the broker nodes. In the experiments, ten topics are predefined. Randomly selected nodes create 20 to 100 unique subscriptions, and 200 to 1000 publications unless stated otherwise. The message creation times are uniformly distributed throughout the experimental duration. The experiment is performed with MIT (100 devices), CAM (40 devices), and UCSD (300 devices) traces. All the results are averaged over at least 5 runs of the experiments.

We have performed experiments of our publish/subscribe communication with several connectivity traces and show the results with three connectivity traces: 1) CAM (well connected nodes in the entire network), 2) MIT (existence of distinct communities), and 3) UCSD (no strong community structure). Table 3, 4, and 5 summarise the results of the *Socio-Aware Overlay* approach. The second column (*Average hops*) is hop counts per publication. The experiment with the MIT trace shows around 1.3 hops regardless of the scale of publication/subscription. The average pair distance of the network is 1.6 hops (see Section 3.4), which indicates that the *Socio-Aware Overlay* approach performs better than flooding to every subscriber by epidemic approach. The total hop count in the entire operation is shown in the final column (*Total Hops*). A pure epidemic approach results in larger hop counts.

In the experiments, communication between brokers is assumed to use direct methods such as access-point WiFi or GPRS. This approach does not need to wait for the next contact with devices to communicate. Thus, if communication between brokers uses unicast or an epidemic approach, *Average hops* will increase. In the experiments, a group of brokers are used instead of a single broker in the community. This requires further work for balancing network work load of brokers and increasing reliability by replication of brokers.



Each publication has three stages during the simulation: (i) a publication is created at time unit (A), (ii) a publisher contacts the other devices to inject its publication to the network at time unit (B), and (iii) the publication is delivered to the subscriber at time unit (C).

The third column (*Contact to Sub*) indicates $C-B$ in the number of time units. The fourth column (*Pub to Sub*) shows that total duration of publishing ($C-A$). A single time unit has a duration of 0.5 seconds, and *Latency* indicates the approximate latency in minutes. Thus, $C-A$ and $C-B$ are indicators of the latency of publications. $C-B$ is much smaller than $C-A$ and $C-A \approx B-A$. On average, it takes over 3 days to get a first contact from when a publication is ready. However, the majority of nodes gets much shorter waiting time until getting a first contact (see Fig. 12). Once the publication is passed to the contacted device, in a few minutes subscribers will receive a publication.

Fig. 12 depicts the distribution of values $C-A$ in three different settings of publication and subscription. The result shows a power law distribution indicating that most event dissemination has short durations. Fig. 13 depicts the distribution of values $C-B$ from publisher's and subscriber's aspects from an experiment with 1000 publications and 100 subscriptions. Certain subscribers (e.g. 70-80) have higher durations, which has various reasons such as that these nodes are away from the centrality nodes in the community (i.e. more than single hop distance), or these nodes may not be part of the community despite them being detected. This will require further investigation.

The value of *Undelivered* indicates the reliability of delivery. The ratio varies from 26% to 58% in 3 settings. The result shown in the last row of Table 4 has the same setting as the first row except publishers and subscribers are in the same communities. When both publishers and subscribers are in the same communities, the *Undelivered* ratio decreases significantly to 3%. In the real world, this may happen frequently as shared interest often creates communities.

Fig. 14 depicts a comparison of two different settings of publishers and subscribers. *MixCommunity* indicates publishers and subscribers are spread across different communities and *WithinCommunity* indicates 90% of both subscribers and publishers of the same topics reside within the same community. Fig. 14a depicts hop counts from publishers to subscribers and shows that topic sharing within communities gives higher reliability with delivery of events in fewer hops. Fig. 14b depicts the distribution of the latency of publications ($C-A$). *MixCommunity* shows high value of latency of the few nodes. Fig. 14b fundamentally presents a power law distribution indicating that the majority of nodes have low latency.

The evaluation of the MIT trace indicates the use of community, and its centrality does much to improve multi-point asynchronous communication. This improves significantly when members of a community share the same topics.

The result from the CAM and UCSD traces in Table 3 and 5 illustrates *Contact to Sub* is 0, which indicates that the messages are delivered to the subscribers as soon as the publisher has a contact with any node. Thus, the network path from the publisher and subscriber exists during the specific time unit. The CAM trace indicates higher reliability of message delivery with shorter latency, which shows that the entire network is well connected. On the other hand, the reliability of message delivery with the UCSD is low, and the latency for the successful delivery is large. The average hop value ≈ 1 indicates the messages must be delivered by the publishers to subscribers when they are in contact. Even though eight communities are uncovered from the UCSD trace, the communities may not be tightly enough connected for supporting our community based approach.

Thus, the experiment results exhibit the MIT traces as the best use of our approach. Further experiments with different scale of traces are in progress.

6. CONCLUSIONS AND FUTURE WORK

We have introduced publish/subscribe communication for PSNs using our novel *Socio-Aware Overlay* in PSNs based on uncovered community structure from human connectivity traces. We have shown an efficient overlay construction for message dissemination by gossipping when devices contact each other. Distributed community detection and messaging overlay construction enable effective multi-point asynchronous communication. The proposed model is simple but powerful when it is applied on DTN/PSN environments. Our approach exploits real human connectivity traces, which provide rich insight into social behaviour. Within communities and societies, there are structures for social networking, and the structure can be powerful for exploiting the information flow. We investigated how the local and global characteristics of the network can be used practically for information dissemination. The research in this area is wide open, and we are working on a series of extension works as described below.

- Explore various centrality characteristics for communication such as use of betweenness centrality for bridging two communities or high degree centrality nodes in the entire network for communications between brokers.
- Integrating spatial/temporal properties of graphs created by device contacts to improve information dissemination and epidemics (e.g. enabling the incorporation of landmark routing).
- Detecting patterns of human behaviour, location/time influenced behaviour, trajectory of groups etc. from the connectivity trace.

Acknowledgement. This research is funded in part by the Haggle project under the EU grant IST-4-027918. We would like to acknowledge Derek Murray for critical reading and constructive comments and the CRAWDAD project [10] for their hosting and sharing of the connectivity/mobility data.

7. REFERENCES

[1] R. Baldoni, R. Beraldi, G. Cugola, M. Migliavacca, and L. Querzoni. Structure-less content-based routing in mobile ad hoc networks. In *Proc. ICPS*, 2005.

[2] S. Berkowitz. *An Introduction to structured analysis: The network approach to social research*. Butteworth, 1982.

[3] A. Carzaniga, D. Rosenblum, and L. Wolf. Design and evaluation of a wise-area event notification service. *ACM Trans. on Computer Systems*, 19(3), 2001.

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

[5] A. Chaintresu, P. Hui, J. Crowcroft, et al. Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding. Technical Report UCAM-CL-TR617, Univ. of Cambridge, 2005.

[6] A. Clauset. Finding local community structure in networks. *Physical Review E*, 72:026132, 2005.

[7] P. Costa, M. Migliavacca, G. P. Picco, and G. Cugola. Epidemic algorithms for reliable content-based publish-subscribe: An evaluation. In *Proc. ICDCS*, pages 552–561, 2004.

[8] L. Danon, J. Duch, A. Diaz-Guilera, and A. Arenas. Comparing community structure identification, 2005.

[9] Dartmouth College. Metrosense project, http://metrosense.cs.dartmouth.edu, 2006.

[10] Dartmouth College. A community resource for archiving wireless data at dartmouth, http://crawdad.cs.dartmouth.edu/index.php, 2007.

[11] C. Diot et al. Haggle Project, http://www.haggleproject.org, 2004.

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

[13] Z. J. Haas, J. Halpern, and L. Li. Gossip-based ad-hoc routing. In *Proc. IEEE INFOCOM*, 2002.

[14] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray. From molecular to modular cell biology. *Nature*, 402(6761 Suppl), December 1999.

[15] T. Henderson et al. The changing usage of a mature campus-wide wireless network. In *Proc. Mobicom*, 2004.

[16] Y. Huang et al. Publish/subscribe in a mobile environments. In *Proc. Workshop on MobiDE*, 2001.

[17] P. Hui and J. Crowcroft. Bubble rap: Forwarding in small world dtns in every decreasing circles. Technical Report UCAM-CL-TR684, Univ. of Cambridge, 2007.

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

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

[20] P. Hui, E.Yoneki, S. Chan, and J. Crowcroft. Distributed community detection in delay tolerant networks. In *Proc. MobiArch*, 2007.

[21] IETF. *Delay Tolerant Network Research Group (DTNRG)*. http://www.dtnrg.org, 2004.

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

[23] B. Karp and H. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. *Computing and Networking*, pages 243–254, 2000.

[24] S.-J. Lee et al. A performance comparison study of ad hoc wireless multicast protocols. In *Proc. IEEE INFOCOM*, pages 565–574, 2000.

[25] J. Leguay et al. Opportunistic content distribution in an urban setting. In *ACM CHANTS*, 2006.

[26] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. In *Proc. SAPIR*, 2004.

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

[28] T. Nicolai, E. Yoneki, N. Behrens, and H. Kenn. Exploring social context with the wireless rope. In *Proc. Workshop MONET: LNCS 4277*, 2006.

[29] E. Nordstrom, C. Diot, R. Gass, and P. Gunningberg. Experienes from measuring human mobility using bluetooth inquiring devices. In *Proc. MobiEval*, 2007.

[30] E. O'Neill et al. Instrumenting the city: Developing methods for observing and understanding the digital cityscape. In *UbiComp*, pages 315–332, 2006.

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

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

[33] UCLA. Urban sensing project, http://research.cens.ucla.edu/, 2006.

[34] UCSD. Wireless topology discovery project, http://sysnet.ucsd.edu/wtd/wtd.html, 2004.

[35] S. Wasserman and K. Faust. *Social network analysis: Method and applications*. Cambridge University Press, 1994.

[36] E. Yoneki and J. Bacon. An adaptive approach to content-based subscription in mobile ad hoc networks. In *Proc. MP2P*, 2004.

[37] E. Yoneki and J. Bacon. Distributed multicast grouping for publish/subscribe over mobile ad hoc networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, 2005.

[38] E. Yoneki, P. Hui, and J. Crowcroft. Visualizing Community Detection in Opportunistic Networks. In *ACM MobiCom - CHANTS*, 2007.

[39] Z. Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *IEEE Comm. Surveys*, 8:547, 2006.