

BUBBLE Rap: Social-based Forwarding in Delay Tolerant Networks

Pan Hui, *Member, IEEE* Jon Crowcroft, *Fellow, IEEE* and Eiko Yoneki, *Member, IEEE*

Abstract—The increasing penetration of smart devices with networking capability form novel networks. Such networks, also referred as Pocket Switched Networks (PSNs), are intermittently connected and represent a paradigm shift of forwarding data in an ad-hoc manner. The social structure and interaction of users of such devices dictate the performance of routing protocols in PSNs. To that end, social information is an essential metric for designing forwarding algorithms for such types of networks. Previous methods relied on building and updating routing tables to cope with dynamic network conditions. On the downside it has been shown that such approaches end up being cost ineffective due to the partial capture of the transient network behavior. A more promising approach would be to capture the intrinsic characteristics of such networks and utilize them in the design of routing algorithms.

In this paper, we exploit two social and structural metrics, namely *centrality* and *community*, using real human mobility traces. The contributions of this paper are two-fold. First we design and evaluate BUBBLE, a novel social-based forwarding algorithm, that utilizes the aforementioned metrics to enhance delivery performance. Second we empirically show that BUBBLE can substantially improve forwarding performance compared to a number of previously proposed algorithms including the benchmarking history-based PROPHET algorithm, and social-based forwarding SimBet algorithm.

Index Terms—Social Networks, Forwarding Algorithms, Delay Tolerant Networks, Pocket Switched Networks, Centrality, Community Detection.



1 INTRODUCTION

WE envision a future in which a multitude of devices carried by people are dynamically networked. We aim to build Pocket Switched Networks (PSN) [1], a type of Delay Tolerant Networks (DTN) [2] for such environments. A PSN utilizes contact opportunities to allow humans to communicate without network infrastructure. We propose an efficient data forwarding mechanism over time evolving graphs of the PSN [3], that copes with dynamical, repeated disconnection and re-wiring. With such scenarios, end-to-end delivery through traditional routing algorithms is rarely applicable.

Many MANET and some DTN routing algorithms [4] [5] provide forwarding by building and updating routing tables whenever mobility occurs. We believe this approach is not appropriate for a PSN, since mobility is often unpredictable, and topology structure is highly dynamic. Rather than exchange much control traffic to create unreliable routing structures, which may only capture the “noise” of the network, we prefer to search for some characteristics of the network which are less volatile than mobility. A PSN is formed by people. Hence, social metrics are intrinsic properties to guide data forwarding in such kinds of human networks. Furthermore, if we can detect these social mobility patterns online in a decentralised way, we can apply these algorithms in practice.

In this paper, we focus on two key social metrics: community and centrality. Co-operation binds, but also divides human

society into communities. 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 [6]. This correlated interaction concept also applies to human, so we can exploit this kind of community information to select forwarding paths. Within a community, some people are more popular, and interact with more people than others (i.e., have high *centrality*); we call them hubs. In this paper, we will exploit community and centrality for data forwarding in PSNs.

Methodologically, community detection [7] [8] can help us to uncover and understand local community structure in both off-line mobile trace analysis and online applications, and is therefore helpful in designing good strategies for information dissemination. Freeman [9] 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 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?

To preview our answers to the above questions, we evaluate the impact of community and centrality on forwarding,

- P. Hui is with Deutsche Telekom Laboratories, Ernst-Reuter-Platz 7, 10587, Berlin, Germany
E-mail: pan.hui@telekom.de
- J. Crowcroft and E. Yoneki are with Computer Laboratory, University of Cambridge, William Gates Building, 15 JJ Thomson Avenue, Cambridge, CB3 0FD, UK

and propose BUBBLE, a hybrid algorithm, that selects high centrality nodes and community members of destination as relays. We demonstrate a significant improvement in forwarding efficiency over a number of previously proposed state-of-the-arts algorithms including PROPHET algorithm [5], which uses patterns of movement, rather than the longer term social relationships on which our BUBBLE scheme rests, and the social-based forwarding SimBet algorithm [10]. In a PSN, there may be no *a priori* information. By definition, we are also in a decentralised world without access to infrastructure. Therefore 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 message delivery performance and efficiency compared to prior schemes.

The rest of this paper is structured as follows. We introduce the experimental data sets in Section 2, describe the contact graphs and inferring human communities in Section 3. In Section 4, we examine the heterogeneity in centrality. We show evaluation methodology in Section 5 and results including discussions from Section 6 to 9 followed by the related works in Section 10. Finally, we conclude the paper with a brief summary of contributions in Section 11.

2 EXPERIMENTAL DATASETS

In this paper, we use four experimental datasets gathered by the Hagggle Project¹ over two years, referred to as *Infocom05*, *HongKong*, *Cambridge*, *Infocom06*; one dataset from the MIT Reality Mining Project [11], referred to as *Reality*. Previously, the characteristics of these datasets such as inter-contact and contact distribution have been explored in several studies [12] [13] [14], to which we refer the reader for further background information. These five datasets cover a rich diversity of environments, ranging from busy metropolitan city (*HongKong*) to quiet 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.).
- 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. A remark about the datasets is that the experiments do not have the same granularity and the finest granularity is limited to 120 seconds. This is because of the trade-off between the duration of the experiments and the accuracy of the samplings.

3 INFERRING HUMAN COMMUNITIES

In PSN, the social network could map to the computer network since people carry the computing devices. To answer the question whether communities of nodes are detectable in PSN traces we need community detection algorithms. In this section, we introduce and evaluate two centralised community detection algorithms: *K-CLIQUE* by Palla *et al.* [15] and weighted network analysis (WNA) by Newman [16]. We use these two centralised algorithm to uncover the community structures in the mobile traces. We believe our evaluation of these algorithms can be useful for future traces gathered by the research community.

Many centralised community detection methods have been proposed and examined in the literature (see the recent review papers by Newman [7] and Danon *et al.* [8]). 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 be able to detect this feature. The *K-CLIQUE* method satisfies 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 [15]. On the other hand, (WNA) can work on weighted graphs directly, and does not need thresholding, but it cannot detect overlapping communities [16]. Thus we chose to use both *K-CLIQUE* and WNA; they each have useful features that complement one another.

3.1 Contact Graphs

In order to help us to present the mobility traces and make it easier for further processing, we introduce the notion of *contact graph*. The way we convert human mobility traces into weighted contact graphs is based on the number of contacts and the contact duration, although we could use other metrics. The nodes of the graphs are the physical nodes from the traces, the edges are the contacts, and the weights of the edges are the values based on the metrics specified such as the number of contacts during the experiment. We can measure the relationship between two people by how many times they meet each other and how long they stay with each other. We naturally think that if two people spend more time together or see each other more often, they are in a closer relationship.

1. <http://www.hagggleproject.org>

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
Number of external devices	264	868	11,357	14,036	NA
Number of external contacts	1,173	2,507	30,714	63,244	NA

TABLE 1
Characteristics of the five experimental data sets

First, we find the distribution of contact durations and number of contacts for the two conference scenarios are quite similar. To prevent redundancy, in the later sections we only selectively show one example, in most cases *Infocom06*, since it contains more participants.

Figure 1 and Figure 2 show the contact duration and number of contacts distribution for each pair in four experiments. For the *HongKong* experiment we include the external device because of the network sparseness, but for the other three experiments we use only the internal devices. These contact graphs created are used for the community detection in the following subsections.

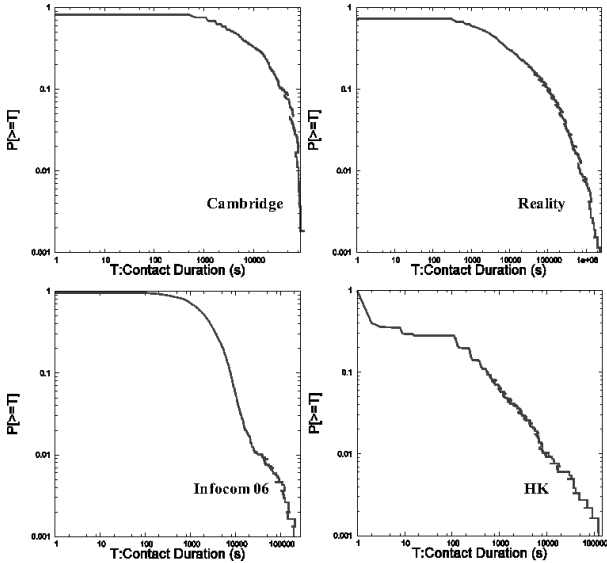


Fig. 1. The distribution of pair-wise contact durations

3.2 K-CLIQUE Community Detection

Palla *et al.* [15] 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. Figure 3 shows the 3-clique communities in the *Infocom06* dataset. More detailed descriptions about the k -clique communities on these datasets can be found in our previous work [17] [18].

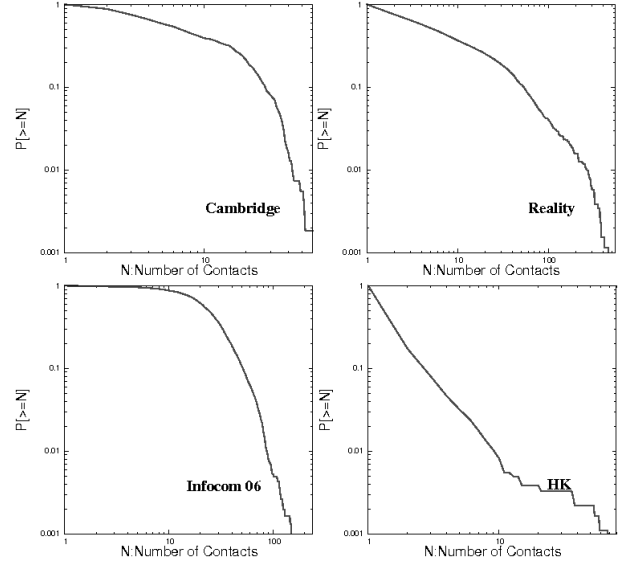


Fig. 2. The distribution of pair-wise number of contacts

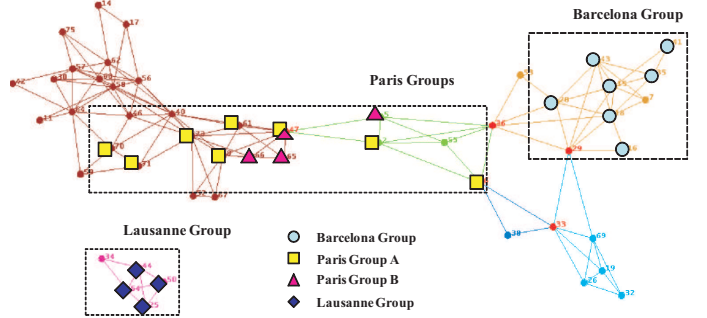


Fig. 3. 3-clique communities based on contact durations with weight threshold that equals 20 000 s (*Infocom06*; circles, Barcelona group; squares, Paris group A; triangles, Paris group B; diamonds, Lausanne group)

3.3 Weighted Network Analysis

In this section, we implement and apply Newman's weighted network analysis (WNA) for our data analysis [16]. This is an extension of the unweighted *modularity* proposed in [19] 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 vertex i belongs to. *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 we keep the degrees of the vertices unchanged. The algorithm is essentially a genetic algorithm, using the modularity as the measurement of fitness. Rather than selecting and mutating current best solutions, we enumerate all possible merges of any two communities in the current solution, and evaluate the relative fitness of the resulting merges, and choose the best solution as the seed for the next iteration.

Table 2 summarises the communities detected by applying WNA on the four datasets. According to Newman [16], 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 spawned 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 2

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.

4 HETEROGENEITY IN CENTRALITY

In human society, people have different levels of popularity: salesmen and politicians meet customers frequently, whereas computer scientists may only meet a few of their colleagues once a year [17]. Here, we want to employ heterogeneity in popularity to help design more efficient forwarding strategies: we prefer to choose popular hubs as relays rather than unpopular ones. SimBet routing algorithm [10] also uses the concept of centrality for choosing relays. We will compare the performance of both algorithms in Section 8

A temporal network or time evolving network is a kind of weighted network. The centrality measure in traditional weighted networks may not work here since the edges are not necessarily concurrent (i.e., the network is dynamic and edges are time-dependent). Hence we need a different way to calculate the centrality of each node in the system. Our approach is as follows:

- 1) Carry out a large number of emulations of unlimited flooding with different uniformly distributed traffic patterns created using the *HaggleSim* emulator (Section 5.1).
- 2) 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, where we only count the delivery with the shortest delay.

We call the number calculated above the *betweenness centrality* of this node in this temporal graph. Of course, it can be normalised 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 Freeman centrality [9].

Initially, we only consider a homogeneous communication pattern, with equal likelihood of every destination, and we do not weight the traffic matrix by locality. We then calculate the global centrality value for the whole homogeneous system. Later, we will analyse the heterogeneous system (Section 8).

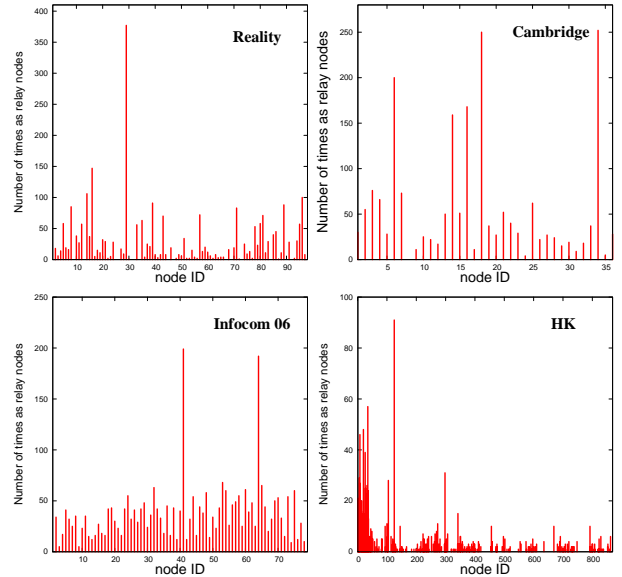


Fig. 4. Number of times a node as relays for others on four datasets

Figure 4 shows the number of times a node falls on the shortest paths between all other node pairs. We can treat this simply as the centrality of a node in the system. We observe 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 that have moderate or low centrality values, across all experiments. One interesting point from the HK data is

that the node showing highest delivery power in the figure is actually an external node. This node could be some popular hub for the whole city, i.e., postman or a newspaper man in a popular underground station, which relayed a certain amount of cross city traffic. The 30th, 70th percentiles and the means of normalised individual node centrality are shown in Table 3. These numbers summarise the statistical property of the centrality values for each system shown in Figure 4.

Experimental dataset	30th percentile	Mean	70th percentile
Cambridge	0.052	0.220	0.194
Reality	0.005	0.070	0.050
Infocom06	0.121	0.188	0.221
Hong Kong	0.000	0.017	0.000

TABLE 3

Statistics about normalised node centrality in 4 experiments

5 INTERACTION AND FORWARDING

In the first half of this paper, we have shown the existence of heterogeneity at the level of individuals and groups, in all the mobility traces. This motivates us to consider a new heterogeneous model of human interaction and mobility.

Cliques and Community: we explored the community structures inside different social environments, and found these community structures match quite well with the real underlying social structures.

Popularity Ranking: We observed the heterogeneity for node centralities in both global and local scales. We shall see that popular hubs are as useful in the PSN context as they are in the wireline Internet and in the Web.

From Section 6 to Section 9, we will look at how can we use this information to make smart forwarding decisions. The following three pre-existing schemes provide lower and upper bounds in terms of cost and delivery success. All of these schemes are inefficient because they assume a homogeneous environment. If the environment is homogeneous then every node is *statistically equivalent* (i.e., every node has the same likelihood of delivering the messages to the destination). As we showed in the first half of this paper, the environments and nodes are diverse, and hence all these naive schemes are doomed to have poor performance. We need to design algorithms which make use of this rich heterogeneity.

- **WAIT:** Hold onto a message until the sender encounters the recipient directly, which represents the lower bound for delivery cost. WAIT is the only single-copy algorithm in this paper.
- **FLOOD:** Messages are flooded throughout the entire system, which represents the upper bound for delivery and cost.
- **Multiple-Copy-multiple-hoP (MCP):** Multiple copies are sent subject to a time-to-live hop count limit on the propagation of messages. By exhaustive emulations, the 4-copy-4-hop MCP scheme is found to be the most cost-effective scheme in terms of delivery ratio and cost for all naive schemes among most of the datasets. Hence for fair comparison, we would like to evaluate our algorithms and

the comparison algorithms (e.g. PROPHET and SimBet) against the 4-copy-4-hop MCP scheme in most of the cases.

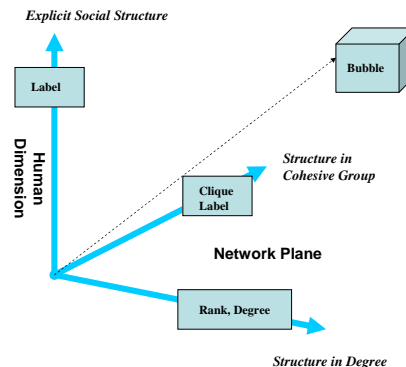


Fig. 5. Design space for forwarding algorithms

Figure 5 shows the design space for the forwarding algorithms. The vertical axis represents the explicit social structure. This is the social or human dimension. The two horizontal axes represent the network structural plane, which can be inferred purely from observed contact patterns. The Structure-in-Cohesive Group axis indicates the use of localised cohesive structure, and the Structure-in-Degree axis indicates the use of node ranking and degree. These are observable physical characteristics. In our design framework, it is not necessary that physical dimensions are orthogonal to the social dimension, but since they represent two different design parameters, we would like to separate them. The design philosophy here is to consider both the social and physical aspects of mobility.

We introduce four forwarding algorithms in this paper, namely LABEL, RANK, DEGREE, and BUBBLE.

- **LABEL:** Explicit labels are used to identify forwarding nodes that belong to the same organisation. Optimisations are examined by comparing label of the potential relay nodes and the label of the destination node. This is in the human dimension, although an analogous version can be done by labelling a k -clique community in the physical domain.
- **RANK:** The forwarding metric used in this algorithm is the node centrality. A message is forwarded to nodes with higher centrality values than the current node. It is based on observations in the network plane, although it also reflects the hub popularity in the human dimension.
- **DEGREE:** The forwarding metric used in this algorithm is the node degree, more specifically the observed average of the degree of a node over a certain time interval. Either the last interval window (S-Window), or a long-term cumulative estimate, (C-Window) is used to provide a fully decentralised approximation for each node's centrality, and then that is used to select forwarding nodes.
- **BUBBLE:** The BUBBLE family of protocols combines the observed hierarchy of centrality of nodes and observed community structure with explicit labels, to decide on the best forwarding nodes. BUBBLE is an example algorithm

which uses information from both human aspects and also the physically observable aspects of mobility.

BUBBLE is a combination of LABEL and RANK. It uses RANK to spread out the messages and uses LABEL to identify the destination community. 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) in the whole system and also a local ranking within its community. It may belong to multiple communities and hence may have multiple local rankings.

5.1 HuggleSim Emulator

In order to evaluate different forwarding algorithms, we developed an emulator called *HuggleSim* [20], which can replay the mobility traces collected and emulate different forwarding strategies on every contact event. This emulator is driven by contact events. The original trace files are divided into discrete sequential contact events, and fed into the emulator as inputs.

In all the simulations in this work, we divided the traces into discrete contact events with granularity of 100 seconds. Our simulator reads the file line by line, treating each line as a discrete encounter event, and makes a forwarding decision on this encounter based on the forwarding algorithm under study.

5.2 Simulation Parameters

There are three parameters we used in our simulation to achieve controlled flooding in MCP strategy.

- *Number of copies (m)*: The maximum number of duplicates of each message created at each node.
- *Number of hops (Hop-TTL)*: The maximum number of hops, counted from the source, that a message copy can travel before reaching the destination; this is similar to TTL value in the Internet.
- *Time TTL*: The maximum time a message can stay in the system after its creation. This is to prevent expired messages from further circulation.

For all the emulations conducted to compare forwarding efficiency in this paper, we have the following two metrics.

- *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 normalise this, we divide it by the total number of unique messages created.

For some cases, we also compute the Hop-count distribution for the deliveries, which is the distribution of the number of hops needed for all the deliveries, and which reveals the social distance between sources and destinations.

In the following sections, we will introduce several forwarding algorithms and evaluate their performances with the above metrics. For each emulation, 1,000 unique messages are created, with the source and destination randomly chosen or chosen based on specific grouping, which will be specified in each evaluation case. The creating time of the messages is

uniformly distributed within the simulation duration. Since the experimental durations for the datasets we used for simulation are in the order of days or even weeks, so we ignore the transient period and average the results throughout the simulation periods.

6 GREEDY RANKING ALGORITHM

The contribution of this section is to introduce RANK algorithm in detail and evaluate its performance using different datasets.

RANK is similar to the greedy strategy introduced by Adamic *et al.* [21]. A PSN is not a static network like the Internet: we do not know when a local maximum is reached since the next encounter is unexpected. We cannot employ precisely the same strategy as they proposed, of traversing up the hierarchy until reaching the maximum, and then down a step. In RANK, we assume each node knows only its own ranking and the rankings of those it encounters, but does not know the ranking of other nodes it does not encounter, and does not know which node has the highest rank in the system. RANK is very simple: we keep pushing traffic on all paths to nodes which have a higher ranking than the current node, until either the destination is reached, or the messages expire.

If a system is small enough, the global ranking of each node is actually the local ranking. If we consider only the Systems Research group (around 40 people), a subset of the Cambridge Computer Laboratory (235 people), this is the ranking of each node inside the group. If we consider the whole Computer Laboratory, we are considering a larger system of many groups, but they all use the same building. A homogeneous ranking can still work. But when we consider the whole city of Cambridge, a homogeneous ranking system would exclude many small scale structures. In this section, we show that in relatively small and homogeneous systems, a simple greedy ranking algorithm can achieve good performance.

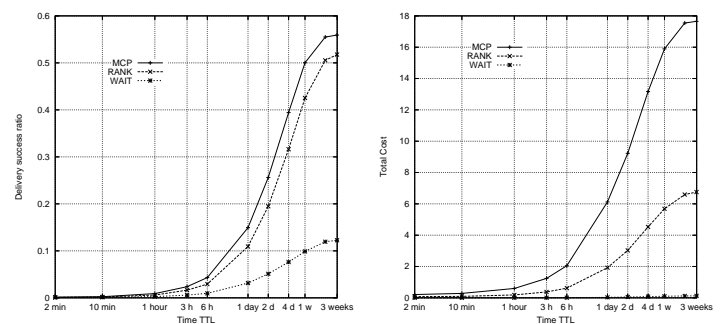


Fig. 6. Comparison of delivery ratio (left) and cost (right) of MCP and RANK on 4 copies and 4 hops case (*Reality*)

Figure 6(left) shows that RANK performs almost as well as MCP for delivery. Figure 6(right) also shows that the cost is at maximum of around 40% that of MCP, which represents a marked improvement. The audiences may notice that the difference in cost between these two algorithms is not constant. This is because they have different spreading mechanisms which may affect their abilities to find necessary number of relays within a certain time TTL.

RANK appears to work in small and homogeneous systems, but when we look at a more diversified system, for example

the *Hong Kong* dataset, it may work differently. In the *Hong Kong* experiment, the 37 participants are intentionally selected without any social correlation. They live and work throughout the whole city. With FLOOD, we can deliver more than 40% of the total traffic across the whole city by using only the 37 iMotes and the external devices detected by these iMotes. But in this case, greedy ranking can only deliver 10% of the messages, although the cost is much lower as well. In terms of delivery and cost, greedy ranking is still more cost-effective than flooding, but clearly the delivery success rate is still too low. One explanation for this low performance is that since the participants have no social correlation, and belong to different social communities, high global ranking of a node may not represent a good choice of relay for some local communities. Messages keep being pushed up to some globally higher ranking nodes, and getting stuck at some maxima, rather than then trickling down to some local community.

Figure 7(a) shows that the maximum number of hops for greedy Rank is 4 hops and after that the messages get stuck. Figure 7(b) shows the rank distribution of the sources, destinations and dead-ends of all the undelivered messages, indicating that message delivery has typically failed at highly-ranked nodes. This supports our hypothesis concerning the dilemma of the messages getting stuck at maxima.

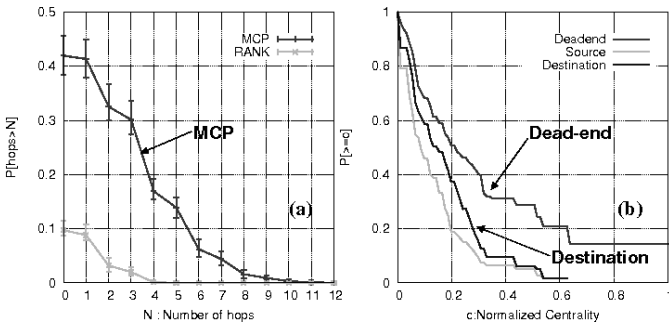


Fig. 7. The hop distribution of the delivered (left) and the rank distribution of undelivered (right) on HK data

Hierarchical organisation is a common feature of many complex systems. The defining feature of a hierarchical organisation is the existence of a hierarchical path connecting any two of its nodes. Trusina *et al.* [22] address how to detect and measure the extent of the hierarchy manifested in the topology of a given complex network. They defined the hierarchical path based on node degrees. A path between two nodes in a network is called hierarchical if it consists of an *up path* where one is allowed to step from node i to node j only if their degrees k_i, k_j satisfy $k_i \leq k_j$, followed by a *down path* where only steps to nodes of lower or equal degree are allowed. Either the up or down path is allowed to have zero length. Because of the good results from the greedy ranking algorithm, we analysed the percentage of hierarchical paths inside all the shortest paths. Table 4 summarises the results.

The percentage of hierarchical paths is calculated as the number of hierarchical paths divided by the number of non-direct deliveries. We can see that for *Cambridge* data and *Reality*, the percentage of hierarchical paths is very high, so our strategy of pushing the messages up the ranking tree

Experimental dataset	% hierarchical paths
Cambridge	87.2 (-2.4,+4.3)
Reality	81.9 (-3.1,+3.3)
Infocom05	62.3 (-2.5,+2.5)
Infocom06	69.5 (-4.1,+2.4)
Hong Kong	33.5 (-4.0,+4.0)

TABLE 4

Hierarchical paths analysis of all shortest paths

can find a lot of these paths, and the performance of the ranking strategy here is not much different from that of MCP. For *Infocom06* and *Infocom05*, the percentage of hierarchical paths is also high, hence the greedy RANK strategy can as well discover many of the shortest paths. However, for the *Hong Kong* experiment, the network is too sparse and a lot of shortest paths are hidden. (This occurs because we could not know the devices detected by the external devices, and most of the resulting paths used for delivery are actually not the shortest) We can see that percentage of hierarchical paths controls the delivery success achieved by the greedy RANK algorithm. We conclude from this that a high percentage of the shortest paths are actually hierarchical paths.

7 DIRECT LABELLING STRATEGY

In the LABEL strategy, each node is assumed to have a label that tells others its affiliation, just like a name badge in a conference. The direct LABEL strategy refers to the exclusively using of labels to forward messages to destinations: next-hop nodes are selected if they belong to the same group (same label) as the destination. It was demonstrated that LABEL significantly improves forwarding efficiency over “oblivious” forwarding using *Infocom06* dataset [20]. This is a beginning of social based forwarding in PSN. The limitation of LABEL is the lack of mechanisms to move messages away from the source when the destinations are socially far away (such as *Reality*). The contribution of this section is to demonstrate the limitations of LABEL strategy, which motivates a new forwarding algorithm using both community and centrality information.

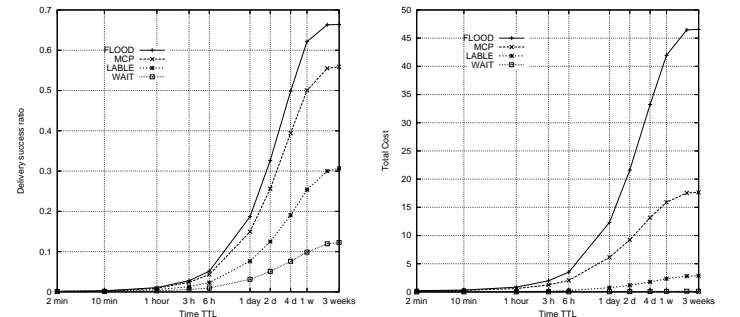


Fig. 8. Comparison of delivery ratio (left) and cost (right) of MCP and LABEL on 4 copies and 4 hops case (*Reality*)

We evaluate the LABEL strategy on the *Reality* dataset. Here we use the community information detected using *K-CLIQUE* algorithm to label the nodes. We can see from Figure 8 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. A person in one group may not meet members in another group so often, so waiting until the members of the other group appear to do the transmission is not effective here.

Figure 9 shows the correlation of the n th-hop relay nodes to the source and destination groups for the messages on all the shortest paths, that is the percentage of the n th-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 same group as 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. We can also see that on going to the 2nd hop, S-Group correlation drops to slightly less than 30%, and when going to 4th-hops, almost all (90%) messages have escaped from this source group. To calculate the percentage for each hop we divide the number of messages which belong to that group (S-Group or D-Group) by the total number of messages destined beyond nodes at that particular hop, but not the total messages created. In the 4-hop case, there are perhaps only 100 messages to forward further, and only 10 out of these 100 relay nodes belong to the source group. This explains why LABEL is not effective, since it is far from discovering the shortest path. In the next section, we will talk about how to use centrality to improve the delivery ratio of LABEL.

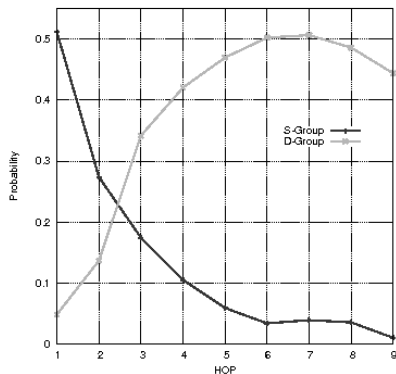


Fig. 9. Correlation of n th-hop nodes with the source group and destination group (*Reality*)

8 CENTRALITY MEETS COMMUNITY

The contribution of this section is to combine the knowledge of both centralities of nodes and community structure, to achieve further performance improvements in forwarding. We show that this avoids the occurrence of the dead-ends encountered with pure global ranking schemes. We call the protocols here BUBBLE, to capture our intuition about the social structure. Messages bubble up and down the social hierarchy, based on the observed community structure and node centrality, together with explicit label data. Bubbles represent a hybrid of social and physically observable heterogeneity of mobility over time and over community.

8.1 Two-community Case

In order to make the study more systematic, we start with the two-community case. We use the *Cambridge* dataset for this study. By experimental design, and confirmed using our community detection algorithm, we can clearly divide the *Cambridge* data into two communities: the undergraduate year-one and year-two group. In order to make the experiment more fair, we limit ourselves to just the two 10-clique groups found with a number-of-contact threshold of 9; that is where each node at least meet another 9 nodes frequently. Some students may skip lectures and cause variations in the results, so this limitation makes our analysis yet more plausible.

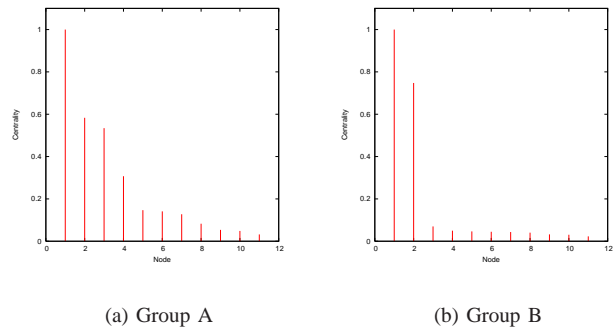


Fig. 10. Node centrality in 2 groups in *Cambridge* data, see Section 4 for the method of calculating the centrality values.

First we look at the simplest case, for the centrality of nodes within each group. In this case, the traffic is created only for members within the same community and only members in the same community are chosen as relays for messages. We can clearly see from Figures 10(a) and 10(b) that inside a community, the centrality of each node is different. In Group B, there are two nodes which are very popular, and have relayed most of the traffic. All the other nodes have low centrality value. Forwarding messages to the popular nodes would make delivery more cost effective for messages within the same community.

Then we consider traffic which is created within each 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 11 shows the individual node centrality when traffic is created from one group to another and the correlation of node centrality within an individual group and inter-group (for data deliveries only to other groups but not to its only 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 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 show now why homogeneous global ranking in Section 6 does not work perfectly. Figure 12 shows the correlation of the local centrality of Group A and the global

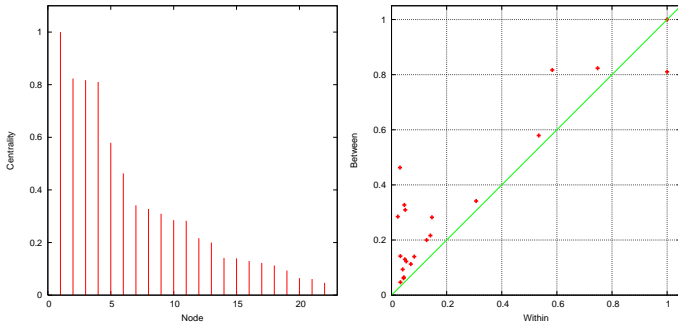


Fig. 11. Inter-group centrality (left) and correlation between intra- and inter-group centrality (right), *Cambridge*

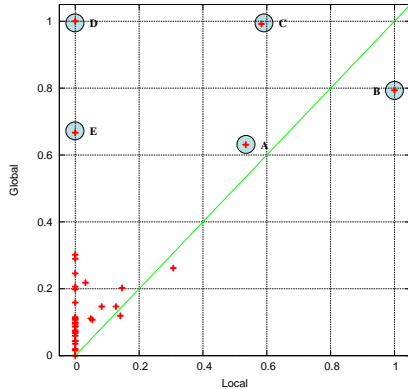


Fig. 12. Correlation of local centrality of group A and the global centrality (*Cambridge*)

centrality of the whole population. We can see that quite a number of nodes from Group A lie along the diagonal line. In this case the global ranking can help to push the traffic toward Group A. However the problem is that some nodes which have very high global rankings are actually not members of Group A, for example node D. Just as in real society, a politician could be very popular in the city of Cambridge, but not a member of the Computer Laboratory, so may not be a very good relay to deliver message to the member in the Computer Laboratory. Now we assume there is a message at node A to deliver to another member of Group A. According to global ranking, we would tend to push the traffic toward B, C, D, and E in the graph. If we pushed the traffic to node C, it would be fine, and to node B it would be perfect. But if it push the traffic to node D and E, the traffic could get stuck there and not be routed back to Group A. If it reaches node B, that is the best relay for traffic within the group, but node D has a higher global ranking than B, and would tend to forward the traffic to node D, where it would probably get stuck again. Here we propose the BUBBLE algorithm to avoid these dead-ends.

Forwarding is carried out as follows. If a node has a message destined for another node, this node would first bubble this message up the hierarchical ranking tree using the global ranking until it reaches a node which has the same label (community) as the destination of this message. Then the local ranking system will be used instead of the global ranking and continue to bubble up the message through the local ranking tree until the destination is reached or the message expired. 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. We call this algorithm BUBBLE, since each world/community is like a bubble. Figure 13 illustrates the BUBBLE algorithm and the pseudo code can be found in our previous work [18].

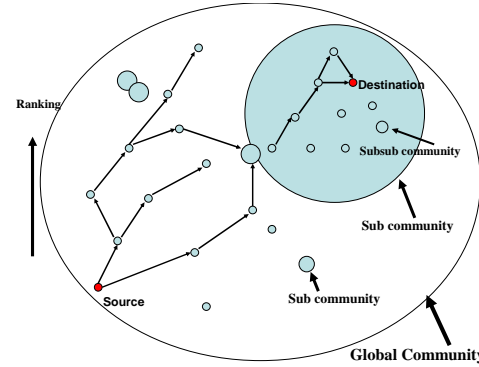


Fig. 13. Illustration of the BUBBLE forwarding algorithm.

This fits our intuition in terms of real life. First you try to forward the data via people more popular than you around you, and then bubble it up to well-known popular people in the society, such as a postman. When the postman meets a member of the destination community, the message will be passed to that community. This community member will try to identify the more popular members within the community and bubble the message up again within the local hierarchy until the message reaching a very popular member, or the destination itself, or the message expires.

A modified version of this strategy is that whenever a message is delivered to the community, the original carrier can delete this message from its buffer to prevent it from further dissemination. This assumes that the community member would be able to deliver this message. We call this protocol with deletion, strategy BUBBLE-B, and the original algorithm introduced above BUBBLE-A.

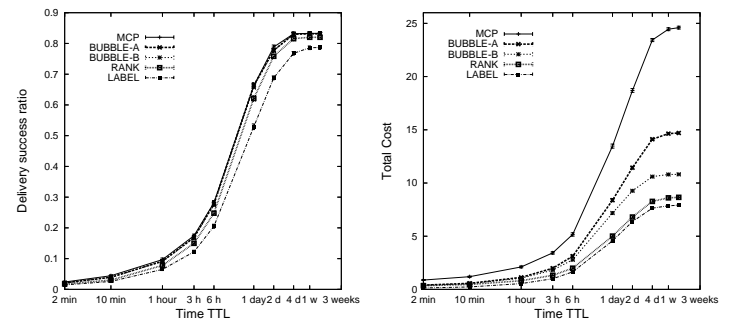


Fig. 14. Comparisons of several algorithms on *Cambridge* dataset

We can see from Figure 14 that both BUBBLE-A and BUBBLE-B achieve almost the same delivery success rate as the 4-copy-4-hop MCP. Although BUBBLE-B has the message deletion mechanism, it achieves exactly the same delivery as BUBBLE-A. BUBBLE-A only has 60% the cost of MCP and BUBBLE-B is even better, with only 45% the cost of MCP. Both have almost the same delivery success as MCP.

8.2 Multiple-community Cases

To study the multiple-community cases, 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 are run over this dataset with uniformly generated traffic.

There is a total of 8 groups within the whole dataset. Figure 15 shows the node centrality in 4 groups, from small-size to medium-size and large-size group. We can see that within each group, almost every node has different centrality.

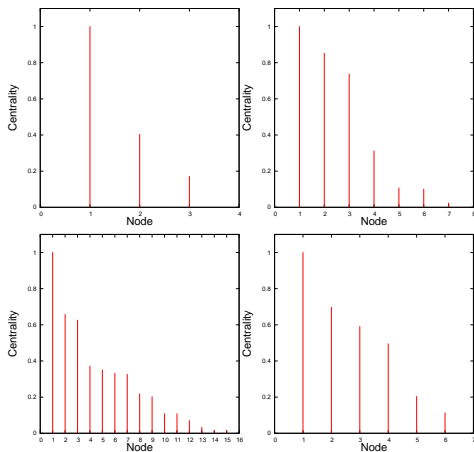


Fig. 15. Node centrality in several individual groups (*Reality*)

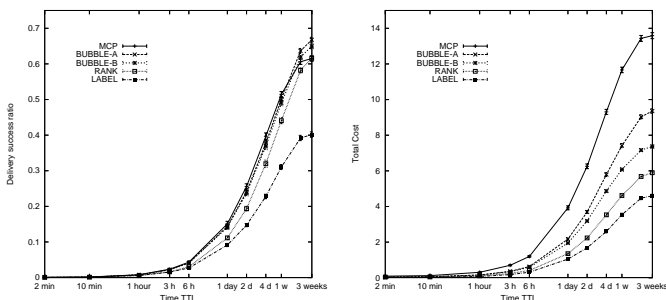


Fig. 16. Comparisons of several algorithms on *Reality* dataset, single group

In order to make our study easier, we first isolate the largest group in Figure 15, 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 16 that BUBBLE-A and BUBBLE-B perform 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 larger than 1 week. BUBBLE-A only has 70% and BUBBLE-B only 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 case of every group creating traffic for other groups, but not for its own members. We want to find the upper cost bound for the BUBBLE algorithm, so we do not consider local ranking (i.e., only global ranking); messages can now be sent to all members in the group. This is exactly a combination of direct LABEL and greedy RANK, using greedy RANK to move the

messages away from the source 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 the BUBBLE algorithms.

From Figure 17, we can see that of course flooding achieves the best performance 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 multiple groups case as well, and even outperforms it when the time TTL of the messages is allowed to be larger than 2 weeks.² However, the cost is only 50% that of MCP. Figure 18 shows the same performance evaluations with the *Infocom06* dataset. In this case, the delivery ratio of RANK is approaching that of MCP but with less than half of the cost. The performance of BUBBLE over RANK is not as significant as in the *Reality* case because in a conference scenario the people are very mixing and hence the community factors are less dominating. We can also see that even in this case, the delivery cost for BUBBLE only increases slightly, which indicates that even in a mixing environment, BUBBLE is still very robust towards the possible misleading of the community factors.

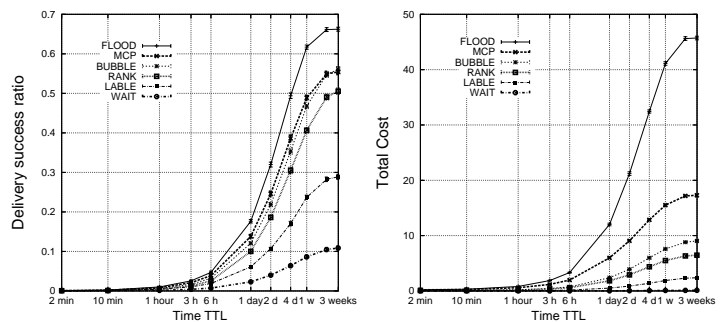


Fig. 17. Comparisons of several algorithms on *Reality* dataset, all groups

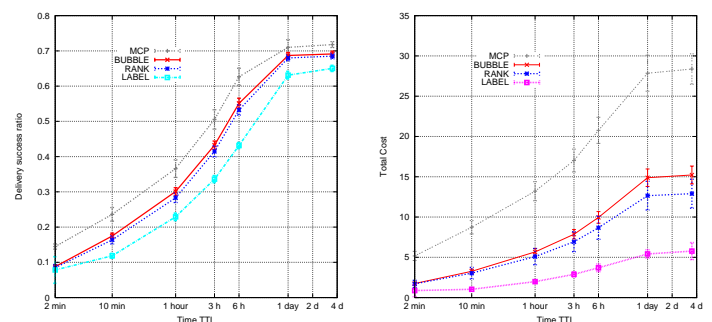


Fig. 18. Comparisons of several algorithms on *Infocom06* dataset, all groups

In order to further justify the significance of social based forwarding, we also compare BUBBLE with a benchmark ‘non-oblivious’ forwarding algorithm, PROPHET[5], and another state-of-the-arts social-based forwarding algorithm, SimBet [10]. PROPHET uses the history of encounters and tran-

² Two weeks seems to be very long, but as we have mentioned before, the *Reality* network is very sparse. We choose it mainly because it has long experimental period and hence more reliable community structures can be inferred. The evaluations here can serve as a proof of concept of the BUBBLE algorithm, although the delays are large in this dataset.

sitivity to calculate the probability that a node can deliver a message to a particular destination. SimBet is similar in concept as BUBBLE for leveraging social contexts. It exploits the exchange of pre-estimate ‘betweenness’ centrality metrics and locally determined social ‘similarity’ to the destination node to guide the message delivery. Since PROPHET has been evaluated against other algorithm before and SimBet is another well-credited social-based algorithm, and both have the same contact-based nature as BUBBLE (i.e., does not need location information), they are good candidates to compare with BUBBLE.

PROPHET has four parameters. We use the default PROPHET parameters as recommended in [5]. 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. For SimBet routing, we use the default values for the SimBet utility parameters as specified in the paper [10] (i.e., $\alpha = \beta = 0.5$) which assigns an equal importance to the similarity and betweenness utility. And since BUBBLE is a multi-copy algorithm, we also implement a multi-copy SimBet for better comparison. For all the comparisons, we use the same settings of number of hops and number of copies for all the algorithms, more particularly we show the results of the 4-hop-4-copy case. The results are also valid for the other combinations of number of hops and number of copies in our simulations. As most of the traces have long experimental durations (i.e. in the order of days or even weeks), we average the results throughout the simulation period and ignore the transient period.

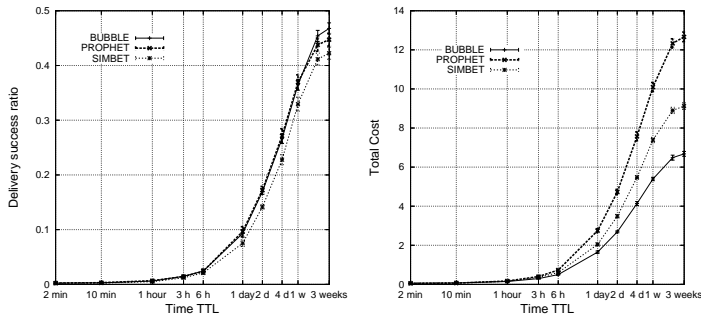


Fig. 19. Comparisons of BUBBLE, PROPHET, and SimBet on Reality dataset

Figures 19 shows the comparison of the delivery ratio and delivery cost of BUBBLE, PROPHET and SimBet for the 4-hop-4-copy case.³ 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 do not count the control traffic created by PROPHET for exchanging routing table during each encounter,

3. For one more perspective on the data, here we show the 4-hop-4-copy case instead of the unlimited case in Figure 17. The trend is very similar for the other case.

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 also do not count the message exchange in SimBet for updating the similarity and betweenness values. We can see that most of the time, BUBBLE achieves a similar delivery ratio to PROPHET and around 10% better than SimBet, but with only half of the cost of PROPHET and 70% of the cost of SimBet. Considering that BUBBLE does not need to keep and update an routing table for each node pairs, the performance achievement is significant.

PROPHET relies on encountering history and transient delivery predictability to choose relays. This can efficiently identify the routing paths to the destinations, but the dynamic environment may result in many nodes having a lot of slightly fluctuation of probabilities. This results in more redundant nodes being chosen as relays, which can be reflected from the delivery cost. Instead, BUBBLE use social information and hence filter out these noises due to the temporal fluctuations of the network. SimBet can successfully leverage social context, but it fails for identifying the sequence of using betweenness and similarity. BUBBLE explicitly identify centrality and community, and first use centrality metric to spread out the messages and then use community metric to focus the messages to the destinations. This approach effectively guarantee a high delivery ratio and a low delivery cost.

A remark here is that the centrality values used for the BUBBLE simulations in this section are calculated in a centralised way, while PROPHET and SimBet use mainly online estimation, but we will show that this can be effectively approximated in a low-cost distributed manner in the next section. Overall, we evaluated BUBBLE against WAIT, FLOOD, the optimised MCP, LABEL, RANK, the benchmark PROPHET, and SimBet. This provides us a reasonable variety of samples to illustrate the performance of BUBBLE.

9 MAKING CENTRALITY PRACTICAL

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. In [23], we have proposed three algorithms, named SIMPLE, K -CLIQUE and MODULARITY, for distributed community detection, and we have proved that detection accuracy can be up to 85% of the centralised K -CLIQUE algorithm. The next step is to ask how can each node know its own centrality in a decentralised way, and how well past centrality can predict the future.

The final contribution of this paper is to provide answers to these two questions.

9.1 Approximating Centrality

We found that the total degree (unique nodes) seen by a node throughout the experiment period is not a good approximation for node centrality. Instead the degree per unit time (for example the number of unique nodes seen per 6 hours, started from midnight) and the node centrality have a high correlation value. We can see from Figure 20 that some nodes with a very high total degree are still not good carriers. It also shows that the per 6 hour degree is quite well correlated to the centrality

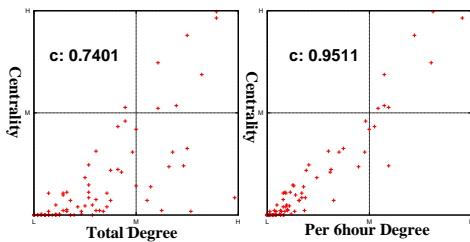


Fig. 20. Correlation of rank with total degree and rank with unit time degree (*Reality*)

value, with correlation coefficient as high as 0.9511. That means how many people you know does not matter too much, but how frequently you interact with these people does matter.

In order to verify that the average unit-time degree is as good as or close to RANK, we run another sets of emulations using greedy average unit-time degree (or we simply call it DEGREE) instead of the pre-calculated centrality. We find that RANK and DEGREE perform almost the same with the delivery and cost lines overlapping each other. They not only have similar delivery but also similar cost.

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 the 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 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 cumulative window (C-Window). This technique is similar to a statistics technique called exponential smoothing [24] and we would like to do further theoretical investigation.

We can see from Figure 21 that the S-Window approach reflects more recent context, and achieves a maximum of 4% improvement in delivery ratio over 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.

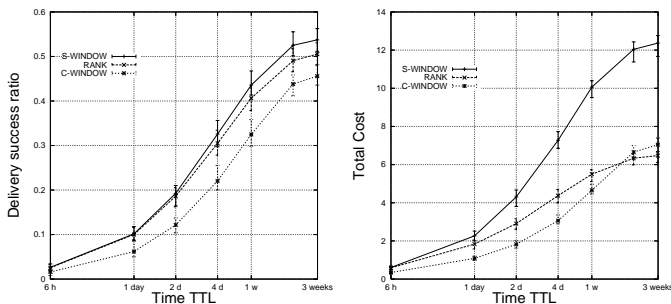


Fig. 21. Comparisons of delivery (left) and cost (right) of RANK, S-Window and C-Window (*Reality*)

9.2 Predictability of Centrality

In order to further verify whether the centrality measured in the past is useful as a predictor for the future, we extracted three temporally consecutive 3-week sessions from the *Reality* dataset and then run a set of greedy RANK emulations on the last two data sessions, but using the centrality values from first session.

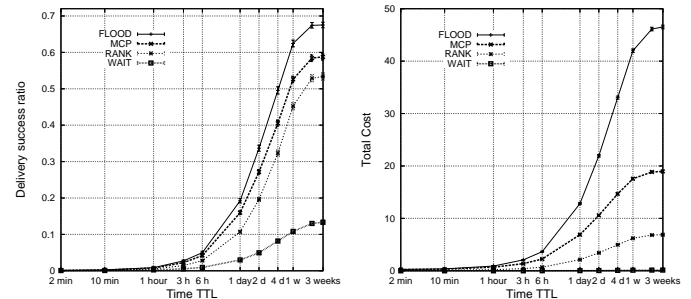


Fig. 22. Delivery ratio (left) and cost (right) of RANK algorithm on 2nd data session, all groups (*Reality*)

Figure 22 shows the delivery ratio and cost of RANK on the 2nd data session using the centrality values from the 1st data session. It seems that the performance of RANK is not far from MCP but with much lower cost, i.e., it is as good as running the emulation on the original dataset which the centrality values derived from. Similar performance is also observed in the 3rd data session. These results imply some level of predictability of human mobility, and show empirically that past contact information can be used in the future.

All these approaches, (DEGREE, S-Window, C-Window and predictability of human mobility) provide us with a decentralised way to approximate the centrality of nodes in the system, and hence help us to design appropriate forwarding algorithms. Combining these approximate methods and the distributed community detection, we can put BUBBLE into reality. We will briefly discuss how distributed BUBBLE works for a city wide environment, but leave the evaluation details as future work when we can get a larger scale of dataset.

Suppose there is a network of mobile users, perhaps spanning an entire city, each device can detect its own local community or knowing its social graph from online social networks [25]. At the same time, it also counts its own 6-hour-averaged degree (i.e., C-Window). Its global ranking can be approximated as its 6-hour-averaged degree for all nodes and its local ranking can be approximated as its 6-hour-averaged degree only for nodes inside its community. With all these metrics, each node can forward messages using BUBBLE. Or we simply call it DiBuBB algorithm, which uses labels from its social graph, affiliation, or distributed community detection for community information and C-Window to approximate its own global and local centrality values. Besides that, it operate exactly like BUBBLE. Figure 23 shows the plotting of delivery ratio against the delivery cost for BUBBLE, DiBuBB, SimBet, and PROPHET. Here, DiBuBB uses the 6-hour C-Window approach to approximate the centrality values and the social graph information for the communities. In general, the larger the slopes of the lines, the more efficient the algorithm is, in term of delivery and cost. We can see that the performance

of DiBuBB is very close to BUBBLE and outperforms both SimBet, and PROPHET.

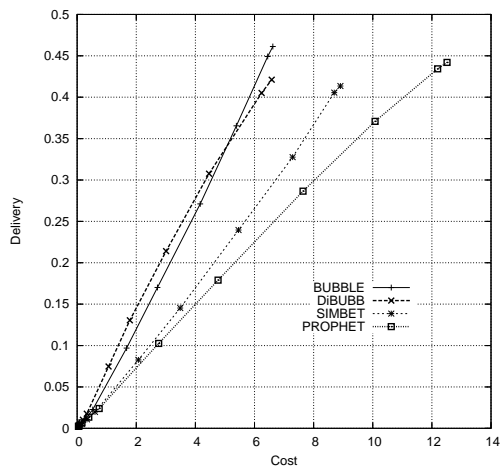


Fig. 23. Delivery ratio against cost for several algorithms, (*Reality*)

10 RELATED WORK

For distributed search for nodes and content in power-law networks, Sarshar *et al.* [26] proposed using a probabilistic broadcast approach: sending out a query message to an edge with probability just above the bond⁴ percolation threshold of the network. They show that if each node caches its directory via a short random walk, then the total number of accessible contents exhibits a first-order phase transition, ensuring very high hit rates just above the percolation threshold.

For routing and forwarding in DTNs and mobile ad hoc networks, there is much existing literature. Vahdat *et al.* proposed epidemic routing, which is similar to the “oblivious” flooding scheme we evaluated in this paper [27]. Spray and Wait is another “oblivious” flooding scheme but with a self-limited number of copies [28]. Grossglauser *et al.* proposed the two-hop relay schemes to improve the capacity of dense ad hoc networks [29]. 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.* [30], location-based routing by Lebrun *et al.* [31], context-based forwarding by Musolesi *et al.* [32] and PROPHET Routing [5] 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 [10] 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. LABEL forwarding [20] uses affiliation information to help forwarding

4. A percolation which considers the lattice edges as the relevant entities.

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. We have compared BUBBLE with LABEL and SimBet in this paper, and demonstrate that by the exploitation of both community and centrality information, BUBBLE provide further improvement in forwarding efficiency. The mobility-assisted Island Hopping forwarding [34] 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.

Finally, we emphasise that we take an experimental rather than theoretical approach, which contrasts with other work described above.

11 CONCLUSION

We have shown that it is possible to uncover important characteristic properties of social network from a diverse set of real world human contact traces. We have demonstrated that community and centrality social metrics can be effectively used in forwarding decisions. Our BUBBLE algorithm is designed for a delay tolerant network environment, built out of human-carried devices, and we have shown that it has similar delivery ratio to, but much lower resource utilisation than flooding, control flooding, PROPHET, and SimBet.

BUBBLE is designed to work better with a hierarchical community structure. The limitation imposed by the size of the datasets (each experiment is not large enough for us to extract hierarchical structure) does not allow us to optimally evaluate it. The current evaluation on a flat community structure did still provide us satisfactory performance improvement. We will further verify our results when more mobility traces are available. Synthetic mobility models can also be useful for further evaluating the algorithm, but currently there is no benchmark models. Another aspect we want to look using our mobility traces is to compare them with the available mobility models and find out the one which can represent most of real mobility scenarios. We believe that this approach represents an early 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.

ACKNOWLEDGMENTS

This research is funded in part by the the Deutsche Telekom Laboratories MADNet project, the ITA project, the EU grants for the Huggle project, IST-4-027918, and the SOCIALNETS project, 217141. 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, Hamed Haddadi, and Georgios Smaragdakis, and SimBet source codes from Thrasylvoulos Spyropoulos.

REFERENCES

- [1] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, “Pocket switched networks and human mobility in conference environments,” in *Proc. WDTN*, 2005.

- [2] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proc. SIGCOMM*, 2003.
- [3] D. Kempe, J. Kleinberg, and A. Kumar, "Connectivity and inference problems for temporal networks," *J. Comput. Syst. Sci.*, vol. 64, no. 4, pp. 820–842, 2002.
- [4] E. P. C. Jones, L. Li, and P. A. S. Ward, "Practical routing in delay-tolerant networks," in *Proc. WDTN*, 2005.
- [5] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," in *Proc. SAPIR*, 2004.
- [6] S. Okasha, "Altruism, group selection and correlated interaction," *British Journal for the Philosophy of Science*, vol. 56, no. 4, pp. 703–725, December 2005.
- [7] M. E. J. Newman, "Detecting community structure in networks," *Eur. Phys. J. B*, vol. 38, pp. 321–330, 2004.
- [8] L. Danon, J. Duch, A. Diaz-Guilera, and A. Arenas, "Comparing community structure identification," *J. Stat. Mech.*, p. P09008, Oct 2005.
- [9] L. C. Freeman, "A set of measuring centrality based on betweenness," *Sociometry*, vol. 40, pp. 35–41, 1977.
- [10] E. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant manets," in *Proceedings of ACM MobiHoc*, 2007.
- [11] N. Eagle and A. Pentland, "Reality mining: sensing complex social systems," *Personal and Ubiquitous Computing*, vol. V10, no. 4, pp. 255–268, May 2006.
- [12] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on the design of opportunistic forwarding algorithms," in *Proc. INFOCOM*, April 2006.
- [13] T. Karagiannis, J.-Y. Le Boudec, and M. Vojnović, "Power law and exponential decay of inter contact times between mobile devices," in *ACM MobiCom '07*, 2007.
- [14] J. Leguay, A. Lindgren, J. Scott, T. Friedman, and J. Crowcroft, "Opportunistic content distribution in an urban setting," in *ACM CHANTS*, 2006, pp. 205–212.
- [15] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005. [Online]. Available: <http://dx.doi.org/10.1038/nature03607>
- [16] M. E. J. Newman, "Analysis of weighted networks," *Physical Review E*, vol. 70, p. 056131, 2004.
- [17] P. Hui and J. Crowcroft, "Human mobility models and opportunistic communications system design," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1872, pp. 2005–2016, June 2008.
- [18] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay tolerant networks," in *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking & computing*, May 2008.
- [19] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, February 2004. [Online]. Available: <http://arxiv.org/abs/cond-mat/0308217>
- [20] P. Hui and J. Crowcroft, "How small labels create big improvements," in *Proc. IEEE ICMAN*, March 2007.
- [21] L. A. Adamic, B. A. Huberman, R. M. Lukose, and A. R. Puniyani, "Search in power law networks," *Physical Review E*, vol. 64, pp. 46 135–46 143, October 2001.
- [22] A. Trusina, S. Maslov, P. Minnhagen, and K. Sneppen, "Hierarchy measures in complex networks," *Physical Review Letters*, vol. 92, p. 178702, 2004. [Online]. Available: [doi:10.1103/PhysRevLett.92.178702](https://doi.org/10.1103/PhysRevLett.92.178702)
- [23] P. Hui, E. Yoneki, S.-Y. Chan, and J. Crowcroft, "Distributed community detection in delay tolerant networks," in *Sigcomm Workshop MobiArch '07*, August 2007.
- [24] P. Winters, "Forecasting sales by exponentially weighted moving averages," *Management Science*, vol. 6, pp. 324–342, 1960.
- [25] P. Hui and N. Sastry, "Real world routing using virtual world information," in *CSE '09: Proceedings of the 2009 International Conference on Computational Science and Engineering*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 1103–1108.
- [26] N. Sarshar *et al.*, "Scalable percolation search in power law networks," June 2004. [Online]. Available: <http://arxiv.org/abs/cond-mat/0406152>
- [27] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Duke University, Tech. Rep. CS-200006, April 2000.
- [28] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and wait: An efficient routing scheme for intermittently connected mobile networks," in *Proc. WDTN*, 2005.
- [29] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad-hoc wireless networks," *IEEE/ACM Trans. on Networking*, vol. 10, pp. 477–486, 2002.
- [30] J. Leguay, T. Friedman, and V. Conan, "Evaluating mobility pattern space routing for DTNs," in *Proc. INFOCOM*, 2006.
- [31] J. Lebrun, C.-N. Chuah, D. Ghosal, and M. Zhang, "Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks," *IEEE VTC*, vol. 4, pp. 2289–2293, 2005.
- [32] M. Musolesi, S. Hailes, *et al.*, "Adaptive routing for intermittently connected mobile ad hoc networks," in *Proc. WOWMOM*, 2005.
- [33] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," in *Proceedings of the MobiCom 2004*, 2004.
- [34] M. P. N. Sarafjanovic-Djukic and M. Grossglauser, "Island hopping: Efficient mobility-assisted forwarding in partitioned networks," in *IEEE SECON*, 2006.



Pan Hui is a senior research scientist in Deutsche Telekom Laboratories, Berlin. He finished his PhD in Computer Laboratory, University of Cambridge in 2007. During his PhD, he also affiliated with Intel Research Cambridge and Thomson Research Laboratory in Paris. Before that he was with University of Hong Kong for his Mphil and bachelor degree. His research interests include delay tolerant networking, mobile networking and systems, planet-scale mobility measurement, social networks, and the application of complex network science in communication system design. More information about his profile and his research work can be found at <http://www.deutsche-telekom-laboratories.de/~panhui/>.



Jon Crowcroft is the Marconi Professor of Communications Systems in the Computer Lab, at the University of Cambridge. Jon graduated in Physics from Trinity College, Cambridge University in 1979, and got an MSc in Computing in 1981, and PhD in 1993 both from UCL. He is a fellow of the ACM, the British Computer Society, the IE[ET] the royal academy of engineering and the IEEE. Jon is a recipient of ACM Sigcomm Award in 2009.



Eiko Yoneki is an EPSRC Research Fellow in the University of Cambridge Computer Laboratory, Systems Research Group. Eiko has received her Ph.D. degree from the University of Cambridge in December, 2006 and a Postgraduate Diploma in Computer Science from the University of Cambridge in 2002. Previously, she has spent several years with IBM (US, Japan, Italy and UK) working on various networking products. Her research interests span distributed systems, networking, and databases over mobile/wireless networks including complex and time-dependent networks. Further details are available at <http://www.cl.cam.ac.uk/~ey204>.