Using Hierarchical Community Structure to Improve Community-Based Message Routing

Matthew Stabeler and **Conrad Lee** and **Graham Williamson** and **Pádraig Cunningham**
School of Computer Science and Informatics
Complex and Adaptive Systems Laboratory
University College Dublin
Dublin 4, Ireland

Abstract

Information about community structure can be useful in a variety of mobile web applications. For instance, it has been shown that community-based methods can be more effective than alternatives for routing messages in delay-tolerant networks. In this paper we present initial research that shows that information on hierarchical structures in communities can further improve the effectiveness of message routing. This is interesting because despite much previous work on the topic, there have been few concrete applications which exploit hierarchical community structure.

Introduction

It is well established that knowledge of community structure is useful for information routing in mobile social networks (Nazir, Ma, and Seneviratne 2009; Hui, Crowcroft, and Yoneki 2010; Vallina-Rodriguez, Hui, and Crowcroft 2009). Information about the community to which the intended recipient of the message belongs provides a larger *target* to guide routing. In this paper we explore the idea that if communities have a hierarchical organization then information about that hierarchical structure can further help message routing. We describe an algorithm that exploits hierarchical community structure for message routing and present a preliminary evaluation that suggests that this algorithm improves routing efficiency.

While earlier research on community-based methods for message routing has emphasized distributed rather than centralized algorithms for community discovery and centrality (Hui, Crowcroft, and Yoneki 2010; Vallina-Rodriguez, Hui, and Crowcroft 2009) we feel that a combined approach is possible, as modern mobile devices have the storage and processing capacity to run the community finding algorithms that have been considered, if a mechanism for sharing the network structure exists.

In the next section we provide an overview of existing research on community-based message routing and present our algorithm, BubbleH, that exploits hierarchical community structure. Next, we present the algorithm we use to uncover hierarchical community structure, H-GCE, and then present an evaluation of the performance of the BubbleH algorithm.

Message Routing

Research into Delay Tolerant Networking (DTN) protocols is concerned with the delivery of messages between nodes within some dynamical, possibly mobile network, in a way that is tolerant to intermittant connections, disconnections and failures. This could be within networks of animals, humans, sensors, satellites or any other system of interacting devices, objects, beings or things. Such systems are often characterised by a sparse network of connections between individuals that change over time. Pocket Switched Networks (PSN) (Hui et al. 2005) which are a sub-set of DTNs, best describe our interests, PSN deal with human networks, specifically, networks created by mobile devices carried by people, for example, mobile phones. In this case, humans interact with each other over time, and their mobile phones communicate via radio antennae (e.g. Bluetooth or WiFi).

There are a number of approaches to delivering messages within DTNs; flooding based approaches, flood the network with copies of messages between any nodes that meet; epidemic-like protocols behave similarly, for example Epidemic Routing (Vahdat and Becker 2000) transmits messages to other nodes with some probability, but limits messages by hop count to reduce overhead. These are perhaps the most effective, but the large number of message copies generated and sent, mean a large overhead in message transmission between nodes. More conservative approaches include Spray and Wait (Spyropoulos, Psounis, and Raghavendra 2005), where a limited number of messages are distributed before a phase where nodes keep the messages until meeting the destination. Other approaches involve probablistic, or opportunistic mechanisms to predict future interactions, such as PROPHET (Lindgren, Doria, and Schelén 2003) and Context-Aware Adaptive Routing (CAR) (Musolesi and Mascolo 2009) which use knowledge about previous contacts to fuel predictions about co-locations which are used to decide next-hop routes.

The measures for success of a DTN algorithm can be encapsulated by three metrics; most often the goal is to have high *delivery ratio*, a low overhead *cost* and low *delivery latency* (Crowcroft et al. 2008).

Community-Based Routing

The BUBBLERap (Hui, Crowcroft, and Yoneki 2010) protocol uses community structure to inform routing decisions,

On node n connection to encountered node p
for all messages m held by n for destination d **do**
  if $p == d$ **then**
    $p \leftarrow m$
  else if $|BC(p,d)| < |BC(n,d)|$ **then**
    $p \leftarrow m$
  else if $BC(p,d) == BC(n,d)$ and $LocalRank(p) > LocalRank(n)$ **then**
    $p \leftarrow m$
  else
    n keeps m
  end if
end for

Figure 1: BubbleH Algorithm, where $BC(p,d)$ is the set of all nodes which represent the smallest community, or *Bridging Community* containing both node p and node d.

it uses K-CLIQUE clustering (Palla et al. 2005) to generate overlapping community structure from a graph formed from contact between nodes. An edge between two nodes is formed when a contact occurs, and the edge weight is incremented with the duration of the contact at every subsequent meeting. Each node is given a global rank, and a rank within each community to which they belong. Ranks are based on their betweenness centrality globally and within each community. To take into account the dynamic nature of contact networks, the authors also threshold edges based on the connected time between nodes, edges where nodes have been connected for less than 4.5 days are removed before the K-CLIQUE algorithm is applied.

Incorporating Community Hierarchy in Routing

Here we present the algorithm that exploits community hierarchical structure, which we call BubbleH. BubbleH is based on the BUBBLERap algorithm (Hui, Crowcroft, and Yoneki 2010), which uses the notion of local and global node ranking to make decisions about message passing. Node rank is based on the betweenness centrality of each node in the global network (global rank) and each community the node belongs to (local rank). As with the BUBBLERap algorithm, we calculate local rankings based on betweenness centrality, however, we do not use global rank. Instead, we use the hierarchy generated by Hierarchical Greedy Clique Expansion (H-GCE, discussed in the appendix) to drive the mobility of messages within the network. When a node encounters another, it considers whether to pass the message on based on how *close* the other node is in the network structure to the destination node.

Figure 1 shows the BubbleH algorithm, in it we refer to a *Bridging Community* ($BC(p,d)$), this is the smallest community that contains the node in question, and the recipient, or destination node. To find the *Bridging Community* between two nodes, we find all of the communities that the nodes share, and pick the shared community that has the lowest member count. In the case where we have multiple candidate communities, we choose the last candidate community found by the H-GCE algorithm. Alternatively, at this point we could also apply further rules for choosing the community, for example; summing the weight of edges within

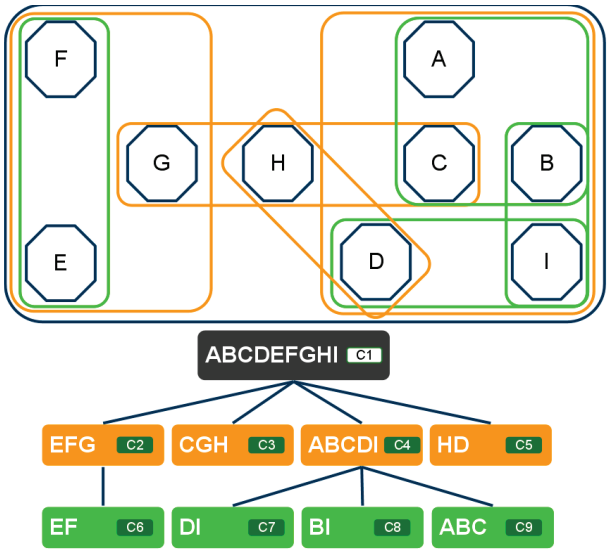

Figure 2: Simplified example of an overlapping hierarchical community structure. The grouping of nodes in the upper section relates to the community hierarchy shown in the lower section. Communities are identified as $C1$ to $C9$, nodes are identified as A to G. For clarity, edges between nodes are not shown, but are weighted by the connected time between the nodes.

the candidate communities and using the highest score or, considering the relative ranking within each candidate community, then picking the community where nodes are *closest* to each other.

To illustrate the concept, Figure 2 shows a simplified overlapping community structure, and its associated hierarchical structure. If we imagine that node F has a message destined for node B, the smallest community containing both F and B is $c1$, this is the *bridging community* for F and B. On encountering another node, F must consider whether the encountered node has a better *bridging community* than itself. For example, when meeting node C, whose *bridging community* with destination node B is $c9$, it will find that $c9$ has less members than $c1$, and pass the message to C.

The intended effect of BubbleH, is to continually narrow down the scope of the message, so that it reaches the smallest community or *target* possible. We believe that in doing so, a message has a better chance of reaching the destination node than, as with BUBBLERap, considering communities of any size without structure.

Hierarchical Community Finding

Any hierarchical community finding method can be plugged into the BubbleH algorithm. Because many community-finding methods are either divisive (e.g., the betweenness-based algorithm introduced in (Newman and Girvan 2004)) or agglomerative (e.g., the efficient modularity-maximization algorithm proposed in (Blondel et al. 2008)), there is no shortage of algorithms that can produce a dendrogram as an output. We now discuss two considerations—community overlap and the significance of hierarchical

structure—which guided our choice of community detection method.

It has recently been observed that mobile phone users typically belong to several network communities, so a suitable community finding algorithm should be capable of detecting overlapping communities. (Ahn, Bagrow, and Lehmann 2010) We therefore require any community finding algorithm we use for BubbleH to be capable of detecting overlapping communities.

Additionally, it is desirable for the hierarchical struture used by BubbleH to be "significant," i.e., an inherent part of the community structure rather than an artifact of the community detection method. This second requirement is related to a point that Fortunato made in his recent review of community detection methods (Fortunato 2010). Although most methods, given a network, will invariably detect some optimal clustering, an important question remains open:

> But is the optimal clustering also *significant*, i.e., a relevant feature of the graph, or is it just a byproduct of randomness and basic structural properties like, e.g., the degree sequence?

This problem of significance looms larger when the task is to find hierarchical community structure. For although some community detection algorithms return a value which indicates the significance of the found communities (such as the modularity Q), very few of the methods which produce a dendrogram also return information on which levels of that hierarchy are significant. The methods which do produce such values are specified to find only non-overlapping communities (Clauset, Moore, and Newman 2008; Rosvall and Bergstrom 2010).

For this reason, we developed H-GCE, a hierarchical community detection method based on GCE (Greedy Clique Expansion), which has been shown to perform well in networks with high levels of community overlap (Lee et al. 2010; Gregory 2011). In addition to producing a dendrogram of overlapping community structure, it labels each point in that dendrogram with a sigfinicance value. The significance value is based on the idea that significant structure is robust to minor random pertubations to the graph structure, and is inspired by bootstrap resampling. For details on the algorithm, see the appendix.

Preliminary Evaluation

To evaluate the performance of BubbleH, we have used ContactSim, a discrete time event simulator designed specifically for evaluating DTN routing. ContactSim is capable of using contact traces either recorded during real-world experiments or generated synthetically. In addition to BubbleH, we have also implemented several other routing algorithms. In order to compare the performance of using hierarchical community structure versus the original K-CLIQUE algorithm, we have implemented BUBBLERap. PROPHET, as one of the most well-known and mature DTN routing protocols, has also been implemented to provide a standard for comparison. Finally, we also show results for unlimited flooding of messages across the network, this gives us bounds on maximum *delivery ratio* and minimum *delivery latency* achievable.

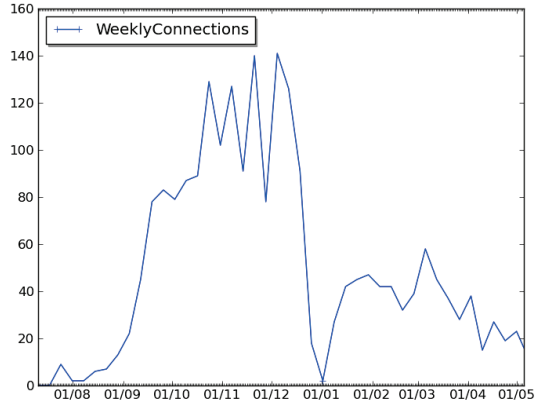

Figure 3: Average weekly Bluetooth connections between participants in the MIT Reality dataset. dataset.

In this experiment, we use Bluetooth proximity traces from the MIT Reality project (Eagle and Pentland 2005) to drive contact events within the simulator. The authors have captured communication, proximity, location, and activity information from 100 subjects at MIT over the course of the 2004-2005 academic year. This data represents over 350,000 hours of continuous data on human behavior.

The MIT Reality dataset has the most number of connections between between Oct 2004 and Jan 2005 (Figure 3), so we chose the period between 10 Nov 2004 and 10 Dec 2004 for community detection and testing, referred to as *MIT-NOV*. A weighted edge list is created from *MIT-NOV*, which is used by both BUBBLERap's thresholded K-CLIQUE and H-GCE to create community structures. For K-CLIQUE the threshold parameter is *30 minutes* which represents 0.07% of the overall time period, with $K = 3$, these parameters are analogous to the optimum settings used in (Hui, Crowcroft, and Yoneki 2010). This creates 6 communities of average size 3 for KCLIQUE, H-CGE creates 27 communities of average size 16. At the start of the simulation, each node is initialized with a message destined for each other node. We used varying parameters to tune H-GCE to find the optimum solution for this network, and have included both the best result (labeled BubbleH), and the average of all of these tuning runs (labeled BubbleH-AVG). The results in Figure 4 show that in this initial evaluation, BubbleH achieves a higher delivery ratio than BUBBLERap, and with a cost that is comparable. PROPHET and BubbleH perform similarly for delivery ratio, but with a much higher cost associated with it. BubbleH-AVG shows an increase in delivery over BUBBLERap, and a similar delivery cost as the best BubbleH and the BUBBLERap run.

Not shown is the latency for BUBBLERap, BubbleH, BubbleH-AVG, PROPHET and Flood, the differences between them appear to be insignificant, apart from Flood, which as expected, gives the best possible latency, a few points better than the other candidates.

Conclusion and Future Work

We have presented methods for identifying hierarchical community structure and routing messages based on this

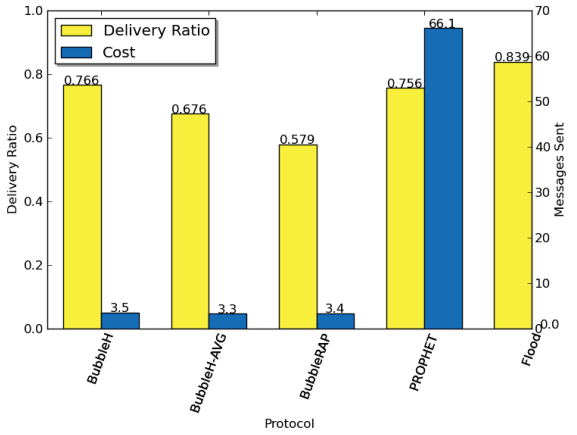

Figure 4: Simulation results for BUBBLERap, BubbleH, PROPHET and Flood during the *MIT-NOV* period.

structure. We have shown that the extra information on hierarchy can improve the efficiency of message delivery. In future work we plan to perform a more comprehensive evaluation to see if these improvements hold up. A key question for the significance of this work is the extent to which hierarchical community structure actually exists in real networks. This is an under-explored question and we plan to test this across a range of real social networks.

Acknowledgments

This work is supported by a PhD scholarship from the Irish Research Council for Science Engineering and Technology and by Science Foundation Ireland Grant No. 08/SRC/I140 (Clique: Graph and Network Analysis Cluster).

Appendix: Detecting hierarchical, overlapping communities using Greedy Clique Expansion

Before describing the algorithm, we define some terms. Let a community consist of a set of nodes. We will call a pair of communities C and C' *near duplicates* if the distance between them $\delta(C, C')$ is less than the value of ϵ, where

$$\delta(C, C') = 1 - \frac{|C \cap C'|}{\min(|C|, |C'|)}. \quad (1)$$

Given a set of communities, we can *merge* two communities by removing them from the set and adding their union.

H-GCE consists of two phases. The first involves identifying a set of seeds and expanding these such that a dendrogram is formed. To find seeds, we first enumerate the set of all cliques in the network, and then remove all cliques which have a larger near-duplicate, as in GCE. Thus, each seed is a clique that is not a near duplicate of a clique of equal or greater size. We then select all seeds of the smallest size and expand them in parallel by one node by greedily optimizing a local fitness function–see (Lee et al. 2010) for details. After each expansion step, we merge any pair of communities which have become near duplicates. This two-step process (expanding all smallest communities by one node, merging any near duplicate pairs) is repeated until each seed has expanded to contain the entire graph. At some point in this phase all seeds will have been merged into one community. Thus, the expansion process produces a singly-rooted tree, or dendrogram, with seeds at the leaves and one big community containing all nodes at the root.

In this dendrogram, the path from a leaf (the seed) to the root is the history of nodes that are added either by greedy expansion or by merging. Thus, each point in this path can be thought of as representing a community containing the node added at that step in the expansion and all nodes added previously.

There is a problem with the dendrogram produced in phase one: even if a graph contains no community structure or only flat community structure, the procedure will return a dendrogram indicating hierarchical community structure–however, this dedrogram will be a mere artifact of the algorithm. The second phase of the algorithm determines which points in the dendrogram produced in the first phase represent significant community structure. The fundamental idea behind our approach is that if community structure is significant, then it will be robust against small perturbations in the graph, whereas insignificant communities will not display such stability. In other words, significant communities will be recoverable even if some noise is added to the graph; this idea has been previously developed in (Karrer, Levina, and Newman 2008; Rosvall and Bergstrom 2008). This and other approaches to measuring the significance of communities are outlined in Fortunato's review of community finding methods (Fortunato 2010).

Phase one leaves us with a set of seeds and a dendrogram. The purpose of phase two is to solve the problem just identifiend by labeling the points in this dendrogram as significant or insignificant. Phase two consists of two steps: the first step produces several dendrograms as in phase one but on graphs that have been perturbed by randomly rewiring ten percent of the edges in such a way that the original degree distribution is preserved. In our experiments we create 100 dendrograms on graphs perturbed using the method outlined in (Karrer, Levina, and Newman 2008).

In order to make sure that all graphs contain the same seeds, we did not rewire edges within seeds. Thus, all of these dendrograms share the same leaves and root as the one produced on the unpertubed graph. This means that as one traverses the path from a particular seed seed s to the root, the nth step of the traversal will have a corresponding point in every dendrogram, i.e., the nth parent of seed s.

Because each of these points corresponds to a community (as explained above), the distance of two correponding points in two dendrograms can be calculated using the distance measure in eq. 2. We call two corresponding points similar if the communities represented by these points have a distance less than 0.1.[1] The second step of phase two compares each point n in the unperturbed dendrogram with the corresponding points n' in the perturbed dendrograms. If n is similar to n' in at least 90% of the perturbed dendrograms,

[1] As *any* two communities approach the size of the entire graph, they will become similar because they are forced to contain the same nodes. For this reason, we use a different similarity measure for two communities if they both contain more than half the nodes in the graph, based on the proportion of nodes that they both ex-

then it is labeled as significant. This defininition of significance is similar to the one used in (Rosvall and Bergstrom 2008). Finally, H-GCE returns the communities corresponding to all points in the original dendrogram which were labeled as significant in phase two.

References

Ahn, Y.-Y.; Bagrow, J. P.; and Lehmann, S. 2010. Link communities reveal multiscale complexity in networks. *Nature* 466(7307):761–764.

Blondel, V. D.; Guillaume, J.-L.; Lambiotte, R.; and Lefebvre, E. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008(10):P10008.

Clauset, A.; Moore, C.; and Newman, M. E. J. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature* 453(7191):98–101.

Crowcroft, J.; Yoneki, E.; Hui, P.; and Henderson, T. 2008. Promoting tolerance for delay tolerant network research. *SIGCOMM Comput. Commun. Rev.* 38(5):63–68.

Eagle, N., and Pentland, A. S. 2005. CRAWDAD data set mit/reality (v. 2005-07-01). Downloaded from http://crawdad.cs.dartmouth.edu/mit/reality.

Fortunato, S. 2010. Community detection in graphs. *Physics Reports* 486(3-5):75–174.

Gregory, S. 2011. Fuzzy overlapping communities in networks. *Journal of Statistical Mechanics: Theory and Experiment* 2011:P02017.

Hui, P.; Chaintreau, A.; Scott, J.; Gass, R.; Crowcroft, J.; and Diot, C. 2005. Pocket switched networks and human mobility in conference environments. *Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking - WDTN '05* 244–251.

Hui, P.; Crowcroft, J.; and Yoneki, E. 2010. Bubble rap: social-based forwarding in delay tolerant networks. *IEEE Transactions on Mobile Computing*.

Karrer, B.; Levina, E.; and Newman, M. E. J. 2008. Robustness of community structure in networks. *Physical Review E* 77(4):046119.

Lee, C.; Reid, F.; McDaid, A.; and Hurley, N. 2010. Detecting highly overlapping community structure by greedy clique expansion. In *SNA-KDD 2010*, 33–42. Washington, DC: ACM.

Lindgren, A.; Doria, A.; and Schelén, O. 2003. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE Mobile Computing and Communications Review* 7(3):19.

Musolesi, M., and Mascolo, C. 2009. CAR: Context-Aware Adaptive Routing for Delay-Tolerant Mobile Networks. *IEEE Transactions on Mobile Computing* 8(2):246–260.

clude, as in

$$\delta(C, C') = 1 - \frac{|(G - C) \cap (G - C')|}{\min(|G - C|, |G - C'|)}. \quad (2)$$

Nazir, F.; Ma, J.; and Seneviratne, A. 2009. Time critical content delivery using predictable patterns in mobile social networks. In *Social Mobile Web (SMW'09) at 2009 IEEE International Conference on Social Computing*, 1066–1073. IEEE.

Newman, M., and Girvan, M. 2004. Finding and evaluating community structure in networks. *Physical review E* 69(2):26113.

Palla, G.; Derényi, I.; Farkas, I.; and Vicsek, T. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435(7043):814–818.

Rosvall, M., and Bergstrom, C. 2008. Mapping change in large networks. *arXiv* 1–9.

Rosvall, M., and Bergstrom, C. 2010. Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems. *Arxiv preprint arXiv:1010.0431*.

Spyropoulos, T.; Psounis, K.; and Raghavendra, C. S. 2005. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, WDTN '05, 252–259. New York, NY, USA: ACM.

Vahdat, A., and Becker, D. 2000. Epidemic Routing for Partially Connected Ad Hoc Networks.

Vallina-Rodriguez, N.; Hui, P.; and Crowcroft, J. 2009. Has anyone seen my goose? social network services in developing regions. In *Social Mobile Web (SMW'09) at 2009 IEEE International Conference on Social Computing*. IEEE. 1048–1053.