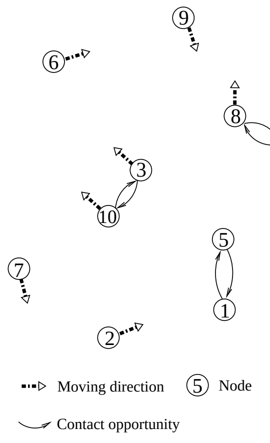
LocalCom: A Community-based Epidemic Forwarding Scheme in Disruption-tolerant Networks

Feng Li and Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431

***Abstract*—In disruption-tolerant networks (DTNs), network topology constantly changes and end-to-end paths can hardly be sustained. However, social network properties are observed in many DTNs and tend to be stable over time. To utilize the social network properties to facilitate packet forwarding, we present LocalCom, a community-based epidemic forwarding scheme that efficiently detects the community structure using limited local information and improves the forwarding efficiency based on the community structure. We define similarity metrics according to nodes' encounter history to depict the neighboring relationship between each pair of nodes. A distributed algorithm, which only utilizes local information, is then applied to detect communities and the formed communities have strong intra-community connections. We also present two schemes to first select and then prune gateways that connect communities to control redundancy and facilitate efficient inter-community packet forwarding. Extensive real-trace-driven simulation results are presented to support the effectiveness of our scheme.**

***Index Terms*—Community detection, disruption-tolerant networks (DTNs), forwarding process, gateways, localized algorithms, social network analysis.**



i	j	(t, Δ)
1	2	(5,40)
	5	**(0,100)**
	7	(25,5)
2	1	(55,40)
	3	(15,30); (70,25)
	10	(20,20); (55,35)
3	2	(15,30); (55,25)
	5	(75,25)
	10	**(15,80)**
4	**8**	**(15,40)**; (60,15)
	9	(5,10); (55,15)
6	7	(15,10); (65,20)
	8	(20,20); (70,20)
8	**4**	**(15,40)**; (60,15)
	5	(70,25)
	6	(20,20); (70,20)
	9	(10,20); (70,20)

Fig. 1. An example of a DTN. The figure on the left shows the instantaneous topology at time 50. In this transient topology, 4 and 8 are mutually connected while 4 and 9 are not connected. The table on the right shows the encounter history recorded by nodes 1 to 4, 6, and 8 in time period [0,100], where t represents start time and Δ represents contact period.

I. Introduction

In disruption-tolerant networks (DTNs), network topology constantly changes and end-to-end paths can hardly be sustained. Information dissemination in DTNs, such as routing, depends on redundancy to reduce latency and increase the delivery ratio. Nodes in DTNs usually send multiple copies to other nodes encountered in order to forward the packet. Uncontrolled duplication and forwarding will always be the best way to achieve the shortest latency and highest delivery ratio. However, constraints such as buffer size and bandwidth always exist and applications usually operate best under a moderate trade-off between latency, delivery ratio, and redundancy.

In order to duplicate and forward the packets in a controlled manner, inherent properties of DTNs are needed to guide the forwarding process. Since mobility is often unpredictable and topology changes can be rapid, traditional topology metrics, such as distance or hop count, are unreliable for the forwarding process. The transient topology can constantly change over time. An example is shown in Fig.1, where the link between 4 and 8 may no longer exist in a subsequent time slot. In this paper, we are interested in utilizing the grouping structure of DTNs to facilitate the forwarding process. After investigating multiple sets of real traces for DTNs [1], [2], we observed that nodes in DTNs tend to meet a certain group of nodes with higher probability compared to with other nodes outside the group, and that the grouping structure remains stable over time. Several recent research works [3], [4], [5] confirm our observation. Using the community structure detected based on global information, a recent forwarding scheme [5] can achieve an acceptable delivery ratio at the cost of one fifth of the total number of forwards to the oblivious flooding under certain conditions. In this paper, we present *LocalCom*, a community-based epidemic forwarding scheme for routing, which efficiently detects the grouping structure using limited local information and utilizes the grouping structure to improve the forwarding efficiency.

To utilize the inherent grouping structure of DTNs, defining a metric to depict the neighboring relationship is the first step. In DTNs, nodes usually have the knowledge of their contacts, also called *encounter history*, which contains both temporal and spacial information. An example encounter history is illustrated in the table in Fig.1. In LocalCom, we select the statistics of the separation period to condense nodes' knowledge into the single metric that we need. A node calculates the average separation period towards its neighbors based on both the frequency and length of their contacts, and applies the Gaussian similarity function [6] to depict the closeness in the relationship. Clearly, a shorter average separation period reflects a closer relationship. Meanwhile, the variance of the separation period is also recorded to reflect the irregularity in the relationship. A single metric called *similarity* can then be deduced from the closeness and irregularity metrics. This metric depicts the relationship between each pair of nodes in the DTN and captures the core of temporal and spacial encounter information. With the similarity metric considering both frequency and duration of contacts, the neighboring graph becomes a novel way to capture spatial and temporal information using the traditional graph structure.

Several community detection schemes [4], [5] have been proposed to design good strategies for information dissemination in DTNs. However, most of them are centralized and focus on analysis of offline mobile traces. In this paper, we develop a distributed scheme in LocalCom that only requires local information to form communities. This scheme uses an *extended clique*, based on virtual links to represent the underlying community structure. These communities have several desirable properties, such as controllable diameter and strong intra-community connection, which can facilitate intra-community communication based on the single-copy source routing.

For inter-community packet forwarding using controlled flooding, we determine the importance of gateways, and use a pruned subset of gateways to forward packets. A *gateway* is a node that has at least one edge in the neighboring graph to nodes in other communities. LocalCom first selects gateways and then prunes them. A static pruning scheme is used together with a dynamic scheme, which keeps gateways of high importance on duty, to effectively trim dispensable gateways (i.e. reducing redundancy) while maintaining the delivery ratio. Both schemes utilize only local information and can be performed in a distributed manner. The trade-off between latency, delivery ratio, and redundancy can be achieved by setting appropriate pruning criteria.

The contributions of this paper are summarized as follows:

1) We present a similarity metric to depict the relationship between each pair of nodes in DTNs. It captures the core of temporal and spacial information in the encounter history and facilities comparison.
2) We design a distributed scheme of community detection. The formed communities show desirable properties for intra-community communication.
3) We propose two pruning schemes which determine the importance of the gateways based on local information and facilitate inter-community communication.
4) We verify the effectiveness of our schemes based on real mobile traces and simulations.

II. Related Work

A. Social network analysis in DTNs

DTNs attempt to route packets via intermittently connected nodes. Vahdat *et al.* proposed epidemic routing [7] which is similar to the oblivious flooding scheme we evaluate in this paper. Spray and wait [8] is another oblivious flooding scheme, but with a self-limited number of copies. MaxProp [9] and PRoPHET [10] both select forwarding nodes based on the nodes' encounter history, and both are examples of how to use system and mobility information to improve the efficiency of forwarding from oblivious flooding.

While early work in DTNs used a variety of simplistic random *i.i.d.* models, such as random waypoint, recent findings [3], [11] show that these models may not be realistic. Moreover, many recent studies [4], [11], [12] based on real mobile traces reveal that DTNs process certain social network properties. Therefore, the social network analysis mechanism is a good tool for determining the properties to improve the forwarding efficiency. Several social network metrics, which are measured based on nodes' direct or indirect observed encounters, are used to guide the packet forwarding in [4]. In [5], [13], Hui *et al.* analyze the community structure from mobility traces and use them for forwarding algorithms, showing a significant improvement in forwarding efficiency. However, with a limited number of hops of local information, neither the community detection nor the weighted network analysis presented in [5], [13] can be used, which restricts the practicality of the methods. In this paper, the LocalCom scheme only needs limited local information to form communities, and achieves comparable forwarding efficiency.

B. Spectral community detection.

In traditional social network analysis, one important step is to identify clusters. *Spectral clustering* [6] is a well studied and widely used centralized clustering mechanism. It usually involves taking the top eigen vectors of some matrix based on the distance between vertices (or other properties) and then using them to cluster the vertexes.

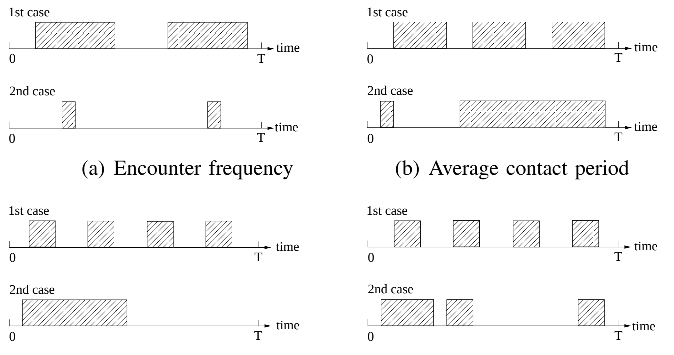
The main tools for spectral clustering are graph Laplacian matrices [14]. The eigenvectors of the graph Laplacian enhance the cluster properties in the data. The core steps of the clustering scheme are: 1) use eigen gap heuristics to determine the number of clusters, 2) change the representation of the graph Laplacian to a matrix that consists of its eigenvectors, and 3) apply the simple k-means clustering algorithm [14] to detect clusters from the eigenvector matrix. We implement a spectral clustering mechanism, which is centralized, as a benchmark method and use it to evaluate the performance and accuracy of LocalCom, a localized scheme.

III. LocalCom Scheme

The LocalCom scheme, which detects and utilizes the inherent social communities to facilitate packet forwarding in DTNs, has three main steps: neighboring graph construction, community detection, and forwarding plan determination.

A. Neighboring graph construction

The key challenge is how to represent a link between two nodes in the DTN with on-off connections. The contact information of a DTN node, which includes both temporal and spacial information, is too complex to be directly used. Therefore, we condense the encounter history associated with



(a) Encounter frequency (b) Average contact period

(c) Total contact period (d) Shortest separation period

Fig. 2. Metrics comparison. A shaded box represents the period that two nodes i, j are within each others' communication range in time interval $[0, T]$.

each edge between two nodes i and j in the DTN into a single *similarity weight* $w_{i,j} \in [0, 1]$. The similarity metric depicts the neighboring relationship and gives hints about the forwarding opportunities between nodes. Larger similarity $w_{i,j}$ indicates a better future contact opportunity between nodes i and j. Possible candidates include encounter frequency, total contact period, average contact period, shortest separation period, and average separation period.

Fig. 2 illustrates the encounter history of nodes i and j. In both cases in Fig. 2(a), there are two shaded boxes, which means that i and j encounter twice in time interval T. The widths of the shaded boxes in the first case are larger than those in the second case, which indicates longer contact periods. Although the encounter frequency is the same, nodes i and j in the first case have better communication opportunities than in the second case. In Fig. 2(b), the average contact periods are the same. However, the first case is preferable since the encounter frequency is higher. In Fig. 2(c), although the total contact periods are the same in both cases, periodic separations enable each node to obtain information from other parts of the network through node movement, and periodic unions enable them to exchange this information. Therefore, the first case still outperforms the second case. In Fig. 2(d), if using shortest separation period metric, the second case should outperforms the first case. However, the first case is still preferable.

Considering Fig. 2, the values of the average separation period comply with the conclusions in all four cases. The average separation period, which is defined as follows, is more comprehensive, since it reflects both the frequency and length of the encounters.

$$AVG(D_{i,j}) = \frac{\int_0^T \delta_{i,j} dt}{n_{i,j}}, \tag{1}$$

where $D_{i,j}$ denotes the separation period between nodes i and j, T is the time elapsed, $n_{i,j}$ represents the number of times that i and j are away from each other, and $\delta_{i,j} = 0$ when i and j are within the mutual communication range; otherwise $\delta_{i,j} = 1$. Smaller $AVG(D_{i,j})$ indicates shorter communication latency between i and j.

We apply the Gaussian similarity function [6] to normalize $AVG(D_{i,j})$ as follows and denote the resulting metric as *closeness* C in this paper:

$$C_{i,j} = e^{-\frac{(AVG(D_{i,j}))^2}{2\sigma^2}}. \tag{2}$$

Here, σ is a scaling parameter [6] for the separation period.

The fluctuation in the separation period should not be neglected. If two cases have the same average separation period, the one with larger fluctuations would be less preferable since the node would be more uncertain regarding the estimated future separation period. Thus, we also need to measure the variance of the separation period distribution to reflect the fluctuation using irregularity metric $I_{i,j}$ as follows:

$$I_{i,j} = VAR(D_{i,j}) = \frac{\sum_l (X_l - AVG(D_{i,j}))^2}{n_{i,j}}, \tag{3}$$

where X_l is the length of separation period l.

We model the neighboring graph of a DTN as $G = (V, E)$, where each vertex i in V corresponds to a node in the DTN; each edge $< i, j >$ in E represents that two nodes have encountered before, and it is associated with a single-weight metric $w_{i,j}$ which is converted from $(C_{i,j}, I_{i,j})$ to facilitate the subsequent community detection and gateway pruning process. There are two possible ways of conversion: one is to view the irregularity metric as a penalty on the closeness metric. Therefore, $w_{i,j} = C_{i,j} - \alpha \cdot I_{i,j}$, where α is a penalty parameter. α decides the importance of the irregularity metric and it should be sufficiently small. Another way is to filter the neighboring relationship with a certain threshold I_T on the irregularity metric. If $I_{i,j} > I_T$, the neighboring relationship will be considered to be too shaky and thusly discarded. Otherwise, $w_{i,j} = C_{i,j}$. As G is undirected, we have $w_{i,j} = w_{j,i}$. We use d_i to denote the degree of node i, which is the sum of the weight of all edges connecting i: $d_i = \sum_{j \in V, j \neq i} w_{i,j}$.

As an example, the neighboring graph in Fig. 4 is constructed based on the encounter history shown in Fig. 1. Since nodes 5 and 6 never met each other before, $w_{5,6} = 0$. $w_{1,5} = 1$ indicates that nodes 1 and 5 are always connected.

B. Community detection

Based on the neighboring graph, we propose a distributed scheme to identify the underlying communities, and represent the communities with an extended clique.

We adopt normalized cuts, $Ncut$ [6], as the benchmark metric in community detection:

$$Ncut(A_1, ..., A_m) = \sum_{l=1}^{m} \frac{\sum_{i \in A_l, j \notin A_l} w_{i,j}}{\sum_{i \in A_l} d_i} \tag{4}$$

Here A_l $(1 \leq l \leq m)$ denotes the formed communities. The numerator is the sum of the weights on all edges connecting a node in A_l to a node outside A_l. The denominator is the sum of the degree for all nodes in A_l. With a small $Ncut$ value, the DTN will be partitioned into communities such that the

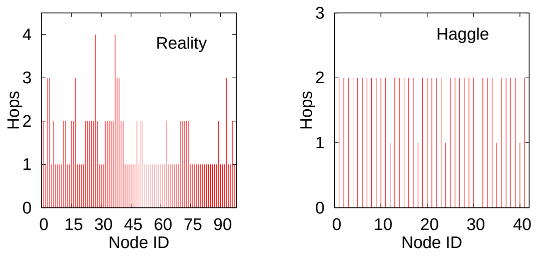


Fig. 3. Max hops to nodes in the same community.

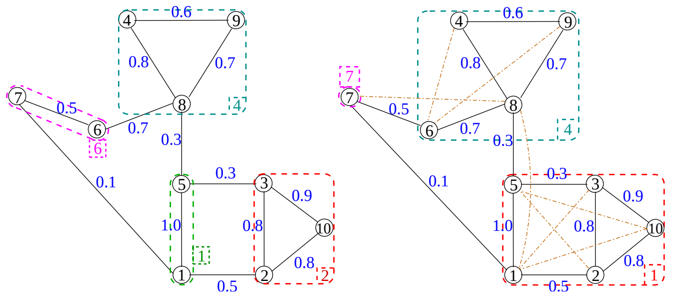

(a) local community with $k = 1$
(b) local community with $k = 2$

Fig. 4. Local community detection, based on Fig. 1. $\sigma = 25$, $w_T = 0.25$. Dashed boxes represents community, and numbers in small squares are the community IDs.

edges between different communities have low weights, the edges within a community have high weights, and the size of the communities is balanced.

In Section II, we mentioned that centralized schemes such as the spectral clustering mechanism require global information. However, the encounter record on nodes constantly changes and the global information is hard to collect. Therefore, we propose a distributed algorithm for community detection. Nodes rely on limited local information to form the communities. Each formed community will be labeled with the smallest node ID among its members' IDs.

Local community definition. In general, the local community is a reflection of locality. A community can be defined based on the notion of clique in graph theory [15]. A *clique* is a subgraph in which every vertex is connected to every other vertex in the graph. If a reasonable threshold w_T is used to filter the neighboring graph, and identify cliques on the filtered graph, each pair of nodes in the clique will have a strong direct neighboring relationship (similarity larger than w_T). However, the original definition for clique is too restrictive for communities in DTNs. Some nodes may have never encountered each other before; therefore, they do not have a direct neighboring relationship. However, they may have a common neighbor that has a close relationship with both. A packet can easily be delivered via the common neighbor.

As case studies, we apply a centralized spectral clustering analysis on the MIT Reality mining [2] and Haggle project dataset [1]. Fig. 3 reflects the diameter of the formed communities. When the hop count for a node i is 2, it indicates that at least one node j exists, which belongs to the same community as i, and that i and j do not have a direct connection but are connected by a 2-hop path in the neighboring graph. As we can see in Fig. 3, many nodes in the same community do not have a direct link. The community should not exclude such multi-hop neighboring relationships, therefore we define a virtual link as follows:

Definition 1: **(Virtual link)** If at least one path with up to k hops between nodes i and j exists, a virtual link can be used to represent i and j's neighboring relationship. The virtual link will be associated with

$$w_{i,j} = \max_{p \in P}\{\prod_{(u,v)\in p} w_{u,v}\}, \tag{5}$$

Here the *path weight* for path p is $\prod_{<u,v>\in p} w_{u,v}$, which is a product of all edge weights along the path. We use the maximum value to represent the virtual link weight as opposed to the sum of the similarity of all paths, because the calculation of the sum of the similarity of all paths does not fit well with single-copy source routing where only one path is selected at a time. An example 2-hop virtual link between nodes 2 and 5 is shown in Fig. 4(b), and the weight is $w_{2,5} = Max\{0.3 \times 0.8, 1.0 \times 0.5\} = 0.5$. Direct links are virtual links with $k = 1$. Based on the concept of the virtual link, we extend the definition of clique and define the local community as the extended clique with virtual links:

Definition 2: **(Local community)** For any pair of nodes in the community, a virtual link exists and the similarity w of the link is larger than the threshold value w_T.

Here, the virtual links are bounded by k. According to the result in Fig. 3, we can use an extended clique with 2-hop virtual links to approximate most of the communities with the size close to the communities detected by the centralized mechanism. Therefore, we consider only $k = 2$ in the following discussion.

Local community detection. Since the communities in real mobility traces tend to have a small diameter, our distributed algorithm aims to divide the nodes into local communities based on limited local information and minimize the normalized cuts. The process consists of three steps: neighborhood information collection, initiator selection, and local community formation.

To form the local community, a node needs to know the virtual links to its neighbors, as well as whether its neighbors are connected by virtual links.

Each node i constructs its neighborhood $N(i)$. Each node in $N(i)$ has a virtual link that connects to node i. Node i also knows $N(N(i))$ (or $N_2(i)$), which includes all the nodes and virtual links in its visible neighborhood.

In LocalCom, a node should select itself as the initiator if its degree d_i is the highest within its visible neighborhood. If two nodes with the same degree are visible to each other, node ID will be used to break the tie.

Each initiator $init$ performs the following steps to form a *local community* with the nodes in $N(init)$:

1) Find a node i, $i \in N(init)$ and $i \notin$ any clique formed by $init$;
2) Start a new clique A with $A = \{init, i\}$.
3) For $\forall j \in N(init)$, add j to A if j has an virtual link with a weight larger than w_T to all the existing nodes in A. Repeat this step until no more nodes can be added to A. After that, calculate $Ncut(A, \bar{A}) = \frac{\sum_{i \in A, j \in \bar{A}} w_{i,j}}{\sum_{l \in A} d_l}$.
4) Repeat steps 1) to 3), until no node i satisfying condition in 1) can be found.
5) Choose the clique with the smallest $Ncut(A, \bar{A})$ value, set the label of this clique as the smallest ID of its members, and inform nodes in $N_2(init)$.

The above heuristic method aims to minimize the normalized cuts in the neighborhood division. Note that the local normalized cut value $Ncut(A, \bar{A})$ is computed based on the weights on the direct links based on the initiator's visible neighborhood. Here $\bar{A}$ denotes nodes in the $N_2(init)$ but outside the community.

Take Fig. 4(b) as an example, where $k = 2$ and $w_T = 0.25$. Node 8 selects itself as the initiator because its degree is the highest in $N_2(8)$. 8 starts a clique with node 4, and adds 6 because 6 has qualified virtual links to both 8 and 4. 9 will be added similarly. Since no node can be further added into $\{8, 4, 6, 9\}$, 8 will go back to step 1) again. This time 8 will start with $\{8, 7\}$ in step 2), and add 6. $\{8, 7, 6\}$ is the new tentative clique. 8 continues this process with 1, and forms $\{8, 1, 5\}$. Finally, 8 will choose $A = \{8, 4, 6, 9\}$ from all formed cliques $\{8, 4, 6, 9\}$, $\{8, 7, 6\}$ and $\{8, 1, 5\}$ in step 5), because it has the smallest $Ncut(A, \bar{A})$. 8 will also propagate the formed community $\{8, 4, 6, 9\}$ with ID 4 to $N_2(8)$.

The initiator should then propagate the formed community to its visible neighbors. Nodes that have not been included in the communities will exclude nodes in the formed communities and continue with the initiator selection and local community formation. We have the following property:

Property 1: **(Effectiveness)** The LocalCom scheme guarantees that nodes in the formed communities have strong intra-community connections, and inter-community connections with low weights.

According to the steps above, each pair of nodes in a formed clique will have a virtual link to connect them, and the weight of the virtual link is larger than w_T. LocalCom also adopts $Ncut(A, \bar{A})$ as the criterion to determine which community should be preserved. Assuming that the source s and destination t of a packet are uniformly chosen from all nodes, and a node randomly chooses one node it encounters as the next forwarder for a packet, we also observe that the probability $P(A|\bar{A})$, which represents the probability that a packet needs to jump into or out of community A, will be reduced since it directly relates to the local $Ncut$ value, $Ncut(A, \bar{A})$. Based on the theory of random walk on graphs [16], we have:

$$P(i \in A, j \in \bar{A}) = \sum_{i \in A, j \in \bar{A}} \frac{d_i}{\sum_{l \in V} d_l} \frac{w_{i,j}}{d_i}.$$

Using this we obtain:

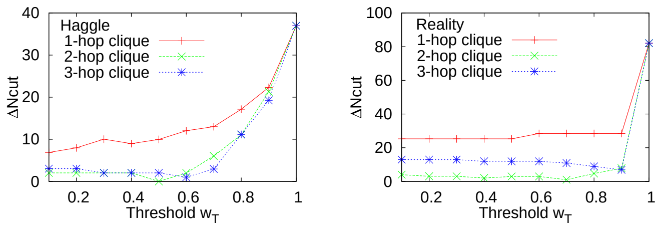


Fig. 5. $Ncut$ value comparison.

$$P(i \in A | j \in \bar{A}) = \frac{P(i \in A, j \in \bar{A})}{P(i \in A)} = \frac{1}{\sum_{l \in A} d_l} \sum_{i \in A, j \in \bar{A}} w_{i,j}.$$

Therefore, $P(A|\bar{A}) = Ncut(A, \bar{A})$. Communities will be determined so that each $P(A|\bar{A})$ is minimized based on the initiator's local view. The result may not be optimal since initiators are selected heuristically and they have a limited local view.

$\Delta Ncut$, which is the difference of $Ncut$ value between the centralized and our localized community detection scheme ($k = 1, 2, 3$), are shown in Fig. 5. Based on both Haggle [1] and Reality [2] datasets, we observe that for the localized community detection scheme with 2-hop extended clique, the $Ncut$ value is as low as the centralized spectral clustering scheme when w_T is around 0.5. This also validates Property 1 according to the definition of $Ncut$ [6]. Fig. 5 also illustrates that a 2-hop extended clique is better than a original clique in both datasets since the $\Delta Ncut$s are significantly lower. The 2-hop extended clique also outperforms the 3-hop extended clique, since it achieves similar $Ncut$ values and requires fewer rounds of information exchange.

C. Forwarding plan determination

Different forwarding plans are adopted for inter- and intra-community packet forwarding in the LocalCom scheme. Since nodes have high similarity and short hop-count distances within the community, nodes should use the single hop source routing for intra-community packet forwarding. The packet will be directly forwarded along a virtual link.

Inter-community communication, which is done through controlled flooding, mainly depends on gateways. *Gateways* are nodes that have direct neighboring relationships with nodes in other communities. Not all gateways are needed in the inter-community forwarding since this may create unnecessary redundancy. In LocalCom, the actual forwarding nodes, called *bridges*, are selected from gateways using two marking and pruning schemes: static pre-pruning and dynamic pruning.

Therefore, if the source and destination nodes of a packet reside in different communities, the source first uses the intra-community forwarding mechanism to forward the packet to the bridges of the current community. The bridges will further forward the packet to other communities they connect to.

Each gateway first conducts pre-pruning based on the static local information. For a node marked to be a bridge in the static pre-pruning, it will further determine its role dynamically based on the information included in a received packet.

Static pre-pruning. Multiple links in the neighboring graph between a pair of communities A and B may exist. Some of them can be pruned according to a designed criterion to achieve a controlled trade-off between latency, delivery ratio, and redundancy. A gateway should mark itself as the bridge if and only if at least one of the inter-community links it connects to is not pruned.

Each gateway calculates its centrality for the community pair A and B it connects. The centrality is a measurement of the structural importance of a node. There are several ways to measure centrality [17]. We choose *betweenness*, which measures how well a node can facilitate communication between two communities in our case. The betweenness centrality of a gateway l in community A connecting the community B is calculated as follows:

$$b_l = \sum_{i \in A} \sum_{j \in B} \frac{\sum_{p \in P_{i,j}(l)} (\prod_{<u,v> \in p} w_{u,v})}{\sum_{p \in P_{i,j}} (\prod_{<u,v> \in p} w_{u,v})}, \tag{6}$$

where $P_{i,j}$ represents the set of all paths between i and j in the neighboring graph, and $P_{i,j}(l)$ denotes the subset of $P_{i,j}$ containing all the paths from i to j that traverse through gateway l. In Equation (6), the numerator is the sum of all path weights in $P_{i,j}(l)$, and the denominator is the sum of all path weights in $P_{i,j}$. A gateway should calculate one distinct centrality value towards each community that it connects to, and send the centrality value to all other nodes in its local community using the virtual links among them.

A gateway $l \in A$ now knows all other gateways in community A connecting to B, and also knows their centrality values. A gateway l calculates a measurement $\bar{w}_l$ to evaluate those gateways, which belong to the same community, have higher centrality than l, and connect the same pair of communities A and B. $\bar{w}_l$ is calculated based on weights and represents the combined forwarding capability of the gateways with higher centrality than l. If the combined forwarding capability satisfies the delivery requirement, l will be able to prune itself to reduce the redundancy without violating the desirable trade-off. More specifically, if both of the following two conditions are satisfied, gateway l prunes itself from being the bridge of the community pair A and B:

1) l's centrality b_l is not the highest among the gateways;
2) For other gateways with higher centrality, the combined similarity $\bar{w}_l$ is equal to or larger than a threshold $\bar{w}_T$.

The combined measurement $\bar{w}_l$ is calculated as follows:

$$\bar{w}_l = \sum_{i=l+1}^{l+m} (w_i \cdot \prod_{j=l+1}^{i-1} (1 - w_j)), \tag{7}$$

where $\{l+1, l+2, ..., l+m\}$ is the set of gateways in A sorted by w_i, and $w_i = \sum_{u \in B} w_{i,u} \cdot \prod_{v \in B, w_{i,v} > w_{i,u}} (1 - w_{i,v})$. A gateway i may have multiple edges connecting with community B, and w_i is the combined measurement of all the edges. The following two properties show the effectiveness of this pre-pruning scheme.

Property 2: **(Connectivity)** If the original neighboring graph is connected, after the static pre-pruning, the reduced graph G' is a connected graph.

The two conditions clearly guarantee connectivity. If a node is the only possible gateway for a connection between two communities, it will be preserved because condition 1) cannot be satisfied. Due to the existence of the centrality order, gateways for the same community level connection will not all be pruned although they make decisions independently. After the pre-pruning process, all the connections between the communities will be preserved. Since nodes in the communities are connected, the connectivity of G' is preserved.

Property 3: **(Controllability)** By adjusting the threshold $\bar{w}_T$, different trade-offs between redundancy and delivery ratio can be achieved.

The pre-pruning scheme provides $\bar{w}_T$ as the parameter to control the trade-off between redundancy and delivery ratio. Only when a gateway l's centrality is low enough and the set of gateways with higher centrality is large enough will l satisfy condition 2). So a higher $\bar{w}_T$ will cause fewer gateways to qualify for condition 2), which leads to a higher delivery ratio and a larger redundancy.

We examine gateway 3 in Fig. 4(a) with the assumption that $\bar{w}_T = 0.5$. Node 3's centrality is 0.38, which is lower than node 2's. Also, $\bar{w}_3 = 0.5 \geq \bar{w}_T$. Therefore, node 3 will be pruned and node 2 will mark itself as the bridge.

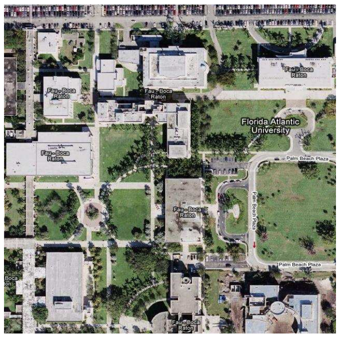
Dynamic pruning. In packet forwarding, each packet can easily record the ID of the last two communities it has traversed. With this piggy backed information, some connections can be dynamically pruned. The bridges for these connections, which are marked in the static pre-pruning, can be pruned.

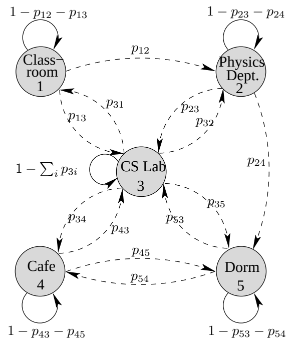
If two communities A and B have links between them on the neighboring graph, we can denote these links together as an edge on the community level. We use $N(A)$ to represent all the communities that are adjacent to A. For example, for community 4 in Fig. 4(b), $N(4) = \{1, 7\}$.

According to the two conditions in static pre-pruning, each community-level connection always consists of at least one bridge after the pre-pruning. If an alternative path on the community level exists that satisfies some properties, one community-level connection would become unnecessary and all gateways for that connection can be pruned.

For a bridge in A which connects B, its collected local information includes $N(A)$, the edges on the community level that connect A and $N(A)$, and those edges between communities in $N(A)$. When receiving a packet for further forwarding, a bridge in A then applies the following pruning process:

1) Mark the edge on the community level e as *necessary*.
2) If e connects community $B \in N(A)$ and B's community ID is recorded in the packet, mark e as *unnecessary*.
3) If e connects community $B \in N(A)$, mark e as the unnecessary edge when an edge on the community level e' exists; e' connects B and $C \in N(A)$, C's community ID is recorded in the packet, and $w_e < w_{e'}$.


(a)



(b)

Fig. 6. The synthetic mobility traces are generated from map of FAU.

TABLE I
Characteristics of three datasets

Dataset	*Haggle*	*Reality*	*Synthetic*
Device	iMotes	Phone	N/A
Network type	Bluetooth	Bluetooth	N/A
Duration (days)	3	246	10
Number of nodes	41	97	100
Number of contacts	$22,459$	$54,667$	Vary

If a bridge is not an end point of a necessary edge, it should prune itself. The following two properties show the effectiveness of the dynamic pruning scheme.

Property 4: **(Correctness)** If the original neighboring graph is connected, all communities remain reachable for packets after the dynamic pruning.

This property can be proved by contradiction. Suppose after we prune one community level connection e, it is the first time that some community becomes unreachable. Also assume e connects A and B and the packet is currently in A. Obviously, B must be a community that becomes unreachable since e has been pruned. However, e can only be pruned when two situations occur: if condition 2) applies, then the packet has reached B, which leads to a contradiction; if condition 3) applies, then the packet has reached C, C and B are connected by edge e', and e' should not be pruned because of e since $w_e < w_{e'}$. Hence, B is reachable for the packet from C through e', which also leads to a contradiction.

Property 5: **(Progressiveness)** Dynamic pruning reduces unnecessary redundancy in inter-community forwarding.

The pruning process actually avoids sending packets back to the community that they originates or when a better alternative edge exists. By applying dynamic pruning, unnecessary forwarding will be avoided and the number of actual bridges will be further reduced.

Take gateway 7 in Fig. 4(a) as an example. Assume it receives a packet containing the community $ID = \{6, 4\}$. Edge $< 1, 7 >$ will be marked as the unnecessary edge because the packet comes from community 4 and $< 5, 8 >$ which also connects to community 1 which has a higher weight. Therefore, gateway 7 will not select itself as a bridge.

IV. Simulation and Analysis

We have conducted simulations to evaluate the effectiveness of the LocalCom scheme in DTNs.

A. Simulation setup

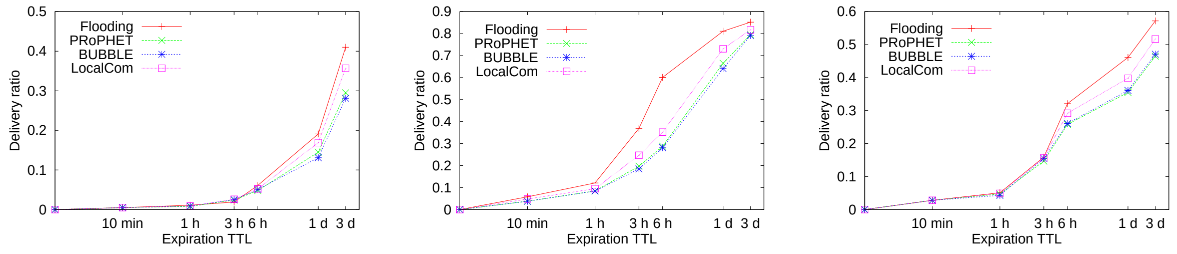
We ran trace-driven simulations with two different datasets: Haggle project [1] and MIT Reality Mining [2]. In Haggle project, 41 iMotes were distributed to students attending Infocom 2005. In Reality, 97 smart phones were deployed to students and staff in MIT. In both datasets, bluetooth contacts were logged and provided. Each contact record includes the start time, end time, and ID of the nodes in contact. For each round of simulation, a portion (default 40%) of the dataset was used as the contact history. The remaining portion is used to evaluate the performance of packet forwarding after the community detection and gateway pruning.

We also adopted a community mobility model proposed in [11], which is considered to be more realistic than *i.i.d.* models. We generated synthetic traces from maps of the Florida Atlantic University (FAU) buildings as shown in Fig. 6 (a). The class schedules and enrollment information of 100 graduate and undergraduate students from three departments were collected. The trace of a node, which represents a network device carried by a student, was generated according to a Markov chain as illustrated in Fig. 6 (b). The states and probabilities were determined by the students' class schedules and enrollment information. If two nodes were in the same building at the same time, they had a probability (default 0.6) to contact each other.

For each simulation, nodes were uniformly selected to be the source or destination of a packet, and $1,000$ packets were generated. All packets had an expiration TTL, which represents the delay requirement. Each node knew only the contact history of itself before the community detection. Each simulation was repeated 30 times with different random seeds for statistical confidence. At each round, default $k = 2$, and $\alpha = 0.1$. We adopted the average separation period of all nodes as σ and the threshold $I_T = (TTL/2)^2$.

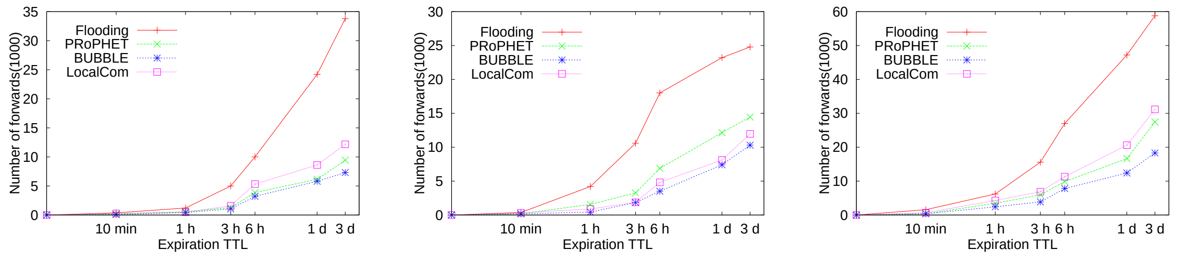
In our simulations, we primarily focused on two parameters: 1) *Delivery ratio:* the proportion of packets that arrived at the destination within the delay requirement; and 2) *Total number of forwards:* the value reflects the overhead in terms of the number of times a packet forward occurred in the DTN.

We now compare the effectiveness of our scheme with three other techniques: simple flooding, PRoPHET [10], and Bubble Rap [5]. In simple flooding, a node copies a packet to every new node it encounters that has not received a copy. PRoPHET is a standard non-oblivious forwarding scheme for DTNs. A node forwards a packet to a node encountered if that node has a higher delivery predictability. Bubble Rap utilizes the community structure based on global encounter history. The centralized scheme can achieve a good delivery ratio at a small cost. To illustrate the effectiveness of our scheme uniformly, we set $w_T = 0.5$ and $\bar{w}_T = 0.8$ in all simulations. These two parameters are actually adjustable and a better result can



(a) Reality (b) Haggle (c) Synthetic

Fig. 7. Performance comparison on successfully delivery ratio.



(a) Reality (b) Haggle (c) Synthetic

Fig. 8. Performance comparison on total number of forwards.

be achieved if these parameters are tuned according to the scenarios in the specific application.

B. Simulation results

As shown in Figs. 7 and 8, the delivery ratio and the total number of packet forwards both increase as the delay requirement on the packet lessens. The delivery ratio and cost of the simple flooding scheme represent the upper bound in all three cases. Since the simple flooding scheme utilizes all the possible paths over time to forward the packet, if a path that can satisfy the delay requirement exists, it will be included. At the same time, the delivery ratio of our scheme outperforms PRoPHET and Bubble Rap in all three scenarios.

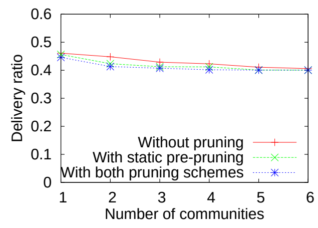
The three datasets represent three different scenarios. The Reality datasets are scenarios that contain many communities and the frequency of contacts are also lower than the other two cases. All four schemes can only achieve a delivery ratio of 30% to 40% when the expiration TTL is three days. The LocalCom scheme achieves a delivery ratio close to the upper bound indicated by the curve for simple flooding, and clearly outperforms PRoPHET and Bubble Rap in this case. The reason for this is that a packet will be broadcast at the community level in LocalCom if the source and destination are in different communities. Therefore, if the packet forwarding is within a community, the source should know the path with the shortest predicted delay; if not, the community level broadcast has a good chance to include the path with the shortest delay. The replication strategy in Bubble Rap is too conservative in this case, therefore it produces a lower delivery ratio as illustrated in Fig. 7(a).

In Fig. 8(a), the total number of forwards in the LocalCom scheme is higher than PRoPHET and Bubble Rap, which is also due to the community level broadcast. The number of selected bridges is high since the number of communities is high in this scenario. Using the local community detection method, eight communities can be detected in the reality dataset. However, the number of forwards is only $1/3$ compared to the simple flooding scheme. Simple flooding is impractical in many applications because the large number of forwards will drain the bandwidth and storage of the DTN.

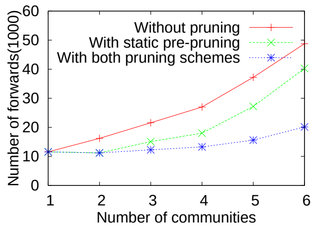
When compared to Bubble Rap, the number of forwards for LocalCom is around 20% more in Fig. 8(a), which is considered acceptable since we only utilize local information, and the delivery ratio is higher in this scenario. The excessive redundant inter-community forwards are effectively reduced since we adopt two gateway pruning methods in our scheme.

The Haggle dataset contains fewer nodes and also fewer communities. Nodes meet more frequently in this dataset. Therefore, the delivery ratio of all four schemes is also higher in Fig. 7(b) than in the other two datasets. When the delay requirement is within three to six hours, our scheme outperforms PRoPHET and Bubble Rap by approximately 15%. The delivery ratios of all schemes are close to each other when the expiration TTL is three days. This indicates that LocalCom can achieve a shorter delay when nodes encounter each other frequently.

In Fig. 8(b), the total number of packet forwards in LocalCom is lower than that of PRoPHET and is close to that of Bubble Rap. The reason for this is because having a small number of communities makes a community level broadcast less costly. Considering this fact together with the delivery ratio, LocalCom shows a clear superiority over other schemes.

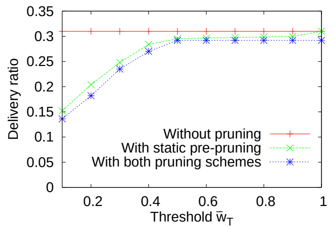


(a) Delivery ratio

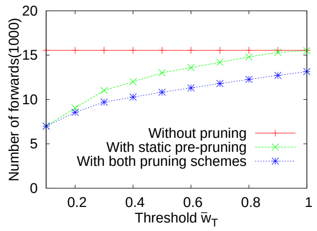


(b) Number of forwards

Fig. 9. Prune strategy comparison using synthetic dataset.



(a) Delivery ratio



(b) Number of forwards

Fig. 10. Controllability by adjusting $\bar{w}_T$.

For the synthetic dataset in Figs. 7(c) and 8(c), we use a moderate number of underlying communities (six detected by the spectral clustering mechanism), and observe contact frequency between that of the Haggle and Reality datasets. Our scheme still shows a clear improvement in delivery ratio towards other schemes except simple flooding, at the cost of a slightly larger total number of forwards.

Figs. 9(a) and 9(b) illustrate the effectiveness of the gateway marking and pruning process. We vary the way that the group of 100 students is constructed in the synthetic dataset to adjust the number of underlying communities, and the expiration TTL is one day. Fig. 9(a) illustrates that both static pre-pruning and dynamic pruning only slightly reduce the delivery ratio. Both methods effectively reduce the number of excessive redundant inter-community forwards as shown in Fig. 9(b).

Figs. 10(a) and 10(b) are based on a synthetic dataset with six communities. The expiration TTL is six hours so that the differences in delivery ratio are sharpened. The results confirm Property 3, a desirable tradeoff between redundancy and delivery ratio that can be achieved by adjusting $\bar{w}_T$. Larger $\bar{w}_T$ causes fewer gateways to be selected in static pre-pruning.

In summation, LocalCom outperforms the centralized social-based scheme Bubble Rap and contact history based scheme PRoPHET in terms of delivery ratio, especially with a moderate delay requirement. Although the total number of forwards of LocalCom is slightly larger than Bubble Rap and PRoPHET, it is significantly lower than that of simple flooding and should be considered acceptable in most applications. Improvement is consistently shown in scenarios with different contact patterns and underlying communities. Considering that nodes only need local information of limited hops to form communities and prune gateways, LocalCom is certainly an efficient distributed forwarding scheme to achieve the desirable tradeoff among latency, delivery ratio, and redundancy.

V. Conclusion

Social network properties are observed in many DTNs and tend to be stable over time. In this paper, we seek to utilize the community structure, which is based on social network properties, to improve routing performance. We define the similarity metrics based on nodes' encounter history to depict the neighboring relationship between nodes. The communities based on real mobility traces tend to show limited diameter by applying a spectral clustering mechanism. We develop a distributed algorithm which only utilizes local information to detect communities. We also present two schemes to mark and prune gateways between communities to achieve configurable trade-offs among latency, delivery ratio, and redundancy. Extensive real-trace-driven simulation results are presented to support the effectiveness of LocalCom. In the future, we plan to study the self-limiting, epidemic routing based on the detected community structure. Nodes will adaptively control the scoping and traffic rates based on their importance and type of communication (intra- or inter-community communication).

References

[1] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. CRAWDAD data set cambridge/haggle (v. 2006-09-15). http://crawdad.cs.dartmouth.edu/cambridge/haggle, September 2006.
[2] N. Eagle and A. Pentland. CRAWDAD data set MIT/reality (v. 2005-07-01). http://crawdad.cs.dartmouth.edu/mit/reality, July 2005.
[3] A. Chaintreau, P. Hui, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, 2007.
[4] E. Daly and M. Haahr. Social network analysis for routing in disconnected delay-tolerant MANETs. In *Proc. of ACM MobiHoc*, 2007.
[5] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: Social-based forwarding in delay tolerant networks. In *Proc. of ACM MobiHoc*, 2008.
[6] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.
[7] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. In *Technical Report CS-200006, Duke University*, 2000.
[8] T. Spyropoulos, K. Psounis, and C. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proc. of ACM SIGCOMM Workshop on DTNs*, 2005.
[9] J. Burgess, B. Gallagher, D. Jensen, and B. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networks. In *Proc. of IEEE INFOCOM*, 2006.
[10] A. Lindgren and A. Doria. Probabilistic routing protocol for intermittently connected networks. *draft-lindgren-dtnrg-prophet-03*, 2007.
[11] W. Hsu, T. Spyropoulos, K. Psounis, and A. Helmy. Modeling time-variant user mobility in wireless mobile networks. In *Proc. of IEEE INFOCOM*, 2007.
[12] N. Djukic, M. Piorkowski, and M. Grossglauser. Island hopping: Efficient mobility-assisted forwarding in partitioned networks. In *Proc. of IEEE SECON*, 2006.
[13] P. Hui, E. Yoneki, S. Chan, and J. Crowcroft. Distributed community detection in delay tolerant networks. In *Proc. of MobiArch*, 2007.
[14] F. Chung. *Spectral Graph Theory.* AMS Press, 1997.
[15] J. Bondy and U. Murty. *Graph theory with applications.* American Elsevier Publishing Company, 1976.
[16] L. Lovasz. Random walks on graphs: a survey. *Combinatorics, Paul Erdos is eighty*, 2:353–397, 1993.
[17] P. Marsden. Egocentric and sociocentric measures of network centrality. *Social Networks*, 24(4):407–422, October 2002.