Up-and-Down Routing in Mobile Opportunistic Social Networks with Bloom-Filter-Based Hints

Huanyang Zheng and Jie Wu
Department of Computer and Information Sciences, Temple University, USA
Email: {huanyang.zheng, jiewu}@temple.edu

***Abstract*—In this paper, an up-and-down routing protocol is proposed for mobile opportunistic social networks, which exhibit a nested core-periphery structure. In such a network, a few active nodes with large weighted degrees form the network core, while the network peripheries are composed of many inactive nodes with small weighted degrees. By nested, it means that the core-periphery structure is preserved, when periphery nodes are removed. Based on this structure, a message can be uploaded from the source to the network core, through iteratively forwarding the message to a relay that has a higher position in the nested network hierarchy. Then, space-efficient Bloom-filter-based hints are introduced to provide guidance for downloading messages from the network core to the destination. Through utilizing the network structure and space-efficient routing hints, subtle balances between the data delivery delay, ratio, and cost are achieved by our proposed approach. Finally, through extensive simulations, we show that the up-and-down routing scheme achieves a competitive performance on the data delivery delay and ratio, with a relatively small cost on the prior information maintenance and a relatively low forwarding cost.**

Keywords*—*Bloom filter, mobile opportunistic social networks, nested core-periphery network structure, routing hints.

I. INTRODUCTION

In recent years, we have witnessed the emergence of a new kind of network known as the *mobile opportunistic social network* (MOSN), where people contact each other by chance using mobile wireless devices. A classic example of an MOSN is a scenario where people walk around with smartphones and communicate with each other via Bluetooth or WiFi when they are in the transmission range of each other [1–3]. Routings in MOSNs are characterized by intermittent network connectivity, where an instantaneous end-to-end path may not exist. Therefore, the prior knowledge on the network state information (e.g., the contact history) is extremely important for improving the MOSN routing performance (e.g., the data delivery delay, ratio, and cost). Meanwhile, the prior state information collection and maintenance are usually difficult and costly due to the highly-dynamical nature of MOSNs.

However, the information of network structure comes handy sometimes for optimizing the routing process. Leskovec et al. found that social networks generally exhibit *nested core-periphery* structures [4], while We observe that MOSNs also exhibit these structures, since wireless devices are actually carried by people. This property is illustrated in Fig. 1: the majority nodes are inactive (small degrees and low contact frequencies), which become the network peripheries (on the left); the minority nodes are active (large degrees and high contact frequencies), which form the network core (on the right). By nested, it means that the core-periphery structure is preserved, when periphery nodes (and their associated links) are removed. Specifically, after iterative removal, the last remaining node is called the *root node*. Furthermore, in Fig. 1, we observe that a few network core nodes in MOSNs hold a large fraction of total contact frequencies, which is the *top-heavy property* [5]. To utilize these MOSN structural properties, we propose an up-and-down routing scheme, which has an *upload phase* and a *download phase*, as follows.

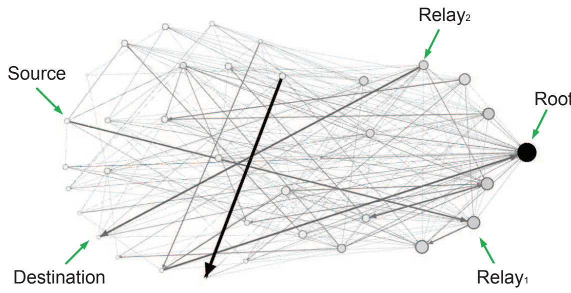


Fig. 1. The contact network in a primary school [6]. Nodes represent people (only nodes labeled "10xx" are selected), where the node with a higher degree is larger and darker. Weighted edges represent contact frequencies, where the edge with a higher weight is thicker and darker.

In the upload phase, a single copy is used to deliver the message from the source node to the network core, by utilizing the nested core-periphery structure, without resorting to the other prior information. This structure enables a message to be uploaded from any node to the root node, through iteratively forwarding this message to a relay that has a higher position in the nested network hierarchy than the message holder. This is done through an iterative labeling process based on weighted degree discounting neighbors that have already labeled. The labeling time of a node determines the hierarchy of the node. This hierarchy is *not* equivalent to the hierarchy used in BubbleRap [7], which performs ranking by node degrees (or weighted degrees). In addition, using only one copy in the upload phase is because it achieves a competitive delivery delay with a low forwarding cost.

A download phase is employed to deliver the message from the network core to the destination. Extra network information is introduced to provide routing guidance. At this stage, if each node maintains an accurate routing table as the extra information, then the delivery delay and ratio are good at the cost of oversized routing tables. To balance the storage demand, we compress the desired network information into Bloom-filter-based hints (maintained by each node respectively), which reduce the hint storage demand at the cost of hint accuracy. To deal with the inaccurate hints, the download phase uses multiple copies to ensure the delivery. The relationship between the hint accuracy and storage demand can be predicted by the top-heavy property. Moreover, Bloom-filter-based hints enable multiple paths for the message download, and thus using multiple copies can effectively reduce the download delay. Meanwhile, *the routing hint update is directional*, which follows the hierarchy used in the upload phase. Therefore, the routing hints have a relatively small maintenance cost. These subtle tradeoffs enable our routing scheme to achieve a good data delivery delay and ratio with a low prior information collection cost and a relatively low forwarding cost.

Our contributions are threefold:

- We systematically explore the MOSN structures, verifying the existences of the nested core-periphery and top-heavy properties. An efficient routing protocol is proposed with the utilization of MOSN properties.
- We study the upload feasibility of network hierarchies. We show that the nested hierarchy has a subtle difference from the degree or weighted degree hierarchies.
- We introduce Bloom-filter-based hints as the download routing guidance to achieve a good data delivery delay and ratio with a low prior information collection cost and a relatively low forwarding cost.

The remainder of the paper is organized as follows: The related work is shown in Section II; the properties of MOSNs are explored in Section III; in Section IV, the space-efficient Bloom-filter-based routing hints are described; in Section V, the up-and-down routing protocol is presented; in Section VI, we evaluate the proposed scheme; and finally, in Section VII, we conclude the paper. All proofs are presented in Appendix.

II. Related Work

Routings in MOSNs have a tradeoff between the routing performance and the prior knowledge on the network state information. The first generation of MOSN routing algorithms, such as Spray and Wait [8], ignores the importance of the prior information. For example, message holders in Spray and Wait share message copies with new encountered nodes that have no copies (spray phase), until only one copy is left for waiting the destination (wait phase). Since no prior information is utilized, the performance of Spray and Wait is much worse than that of the up-to-date algorithms, i.e., larger delivery delay, lower delivery ratio, and higher forwarding cost.

After a short period of time, researchers have realized the importance of the prior information. Therefore, routings with prior information such as contact history [9] or pre-designed hierarchical levels [10, 11] are developed. However, collecting this prior information is very expensive. Researchers have not focused on the network structural information until the emergence of BubbleRap [7], which is a hierarchical approach that uses the structural information. This algorithm uses ranks (i.e., node degrees or weighted degrees) to layer nodes, where the messages are iteratively forwarded to nodes with higher

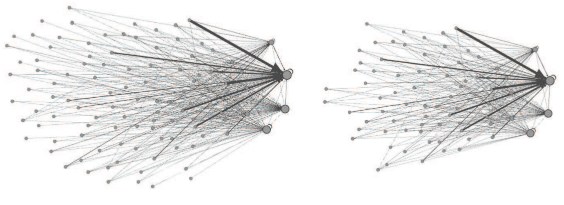

(a) Complete trace. (b) 30% removed.

Fig. 2. The nested core-periphery structures of the MIT trace.

ranks, until a local community that contains the destination is encountered. However, whether the messages will trap in some local high-rank nodes or not remains unknown. To our best knowledge, this paper is the first study on the upload feasibility, which is done through the network structure analysis (we systematically explore the MOSN structures). The nested hierarchy is proposed for the message upload, instead of the degree and weighted degree hierarchies. Meanwhile, BubbleRap tacitly assumes that high-rank nodes are connected to a relay in the same community as the destination, which is not always true among different MOSNs. We will deal with this problem through Bloom-filter-based routing hints.

III. Structures of MOSNs

In this section, we explore structures of small-scale MOSNs that are composed of dozens of people moving around several offices or rooms (e.g., attending an academic conference). These MOSNs are modeled as directed weighted networks. The contact frequency from node i to j is recorded as the link weight [12], which is denoted as λ_{ij}. A larger λ_{ij} means more frequent inter-meetings from node i to node j, and thus a lower average link delay. For simplicity, we assume the links to be symmetric, i.e., $\lambda_{ij} = \lambda_{ji}$. Meanwhile, we define the weighted degree as the sum of the weights associated with every outgoing link incident to the corresponding node:

$$weighted\ degree\,(i) = \sum\nolimits_{j} \lambda_{ij} \tag{1}$$

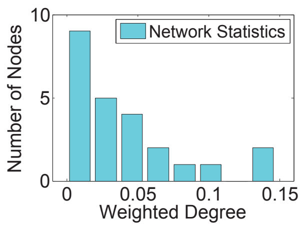
A larger weighted degree means that the corresponding node is more active. For information dissemination in MOSNs, node activities (i.e., weighted degree) are more important than node connectivities (i.e., degree), since messages are delivered through contacts. Note that a weighted edge can be viewed as multiple edges with unit weights. Therefore, in the next two subsections, we will use node activities to verify the nested core-periphery property and the top-heavy property in MOSNs. In the third subsection, we explore the hierarchies of MOSNs.

A. Nested Core-periphery Structures in MOSNs

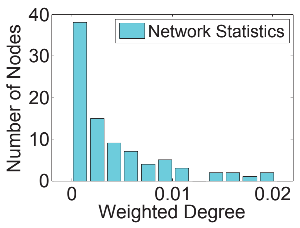
The classic scenario of an MOSN is that people walk around with smartphones and communicate with each other via Bluetooth or WiFi when they are in the transmission range of each other. Since wireless devices are actually carried by people, MOSNs should inherit nested core-periphery structures, in terms of weighted degrees, from social networks [4]. An active person that has a large degree in the social network should have a larger weighted degree in the corresponding MOSN. In other words, the structures of MOSNs are similar to, but are

TABLE I. STATISTICS OF THE MOBILE OPPORTUNISTIC SOCIAL NETWORKS

CRAWDAD Trace	Total Contacts	Duration	Internal Nodes	External Nodes	Network Diameter	The Fraction of Contacts Hold by The Most-active 20% Nodes	Total Number of of Root Nodes
Cambridge/Haggle/Imote/Intel	2,766	6 days	9	128	1	30.72%	1 node
Cambridge/Haggle/Imote/Cambridge	6,732	7 days	12	223	1	51.27%	1 node
Cambridge/Haggle/Imote/Infocom	28,126	4 days	41	264	2	29.83%	1 node
Thlab/Sigcomm2009/Mobiclique/Proximity	285,880	5 days	76	11,938	2	43.64%	1 node
ST_Andrews/Sassy/Mobile	112,265	79 days	27	N/A	2	55.14%	1 node



(a) ST_Andrews trace.



(b) MIT trace.

Fig. 3. Verification of top-heavy properties in MOSNs (two traces).

much denser than structures of social networks. Therefore, the nested core-periphery property also exists in MOSNs.

For further verification, we look into the MIT [13] trace, as shown in Fig. 2 (the same setting as Fig. 1). In this figure, nodes represent people, where a node with a higher weighted degree (a higher node activity) is larger and darker. Weighted edges represent contact frequencies, where the edge with a higher weight is thicker and darker. Then, Fig. 2(a) shows the complete trace (106 nodes), which has a core-periphery structure. Fig. 2(b) shows the partial trace, where 30% nodes with low weighted degrees are removed. Fig. 2(b) also exhibits a core-periphery structure, i.e., the MIT trace has a nested core-periphery structure.

B. Top-heavy Property in MOSNs

In this subsection, we investigate the weighted degree distribution of MOSNs. The ST_Andrews [14] and MIT [13] traces are used to verify the existence of top-heavy properties in MOSNs, the results of which are shown in Fig. 3 (the unit of the weighted degree is the number of contacts per minute). For the ST_Andrews trace, devices with ID 16, 26 and 27 are removed, since no data was collected. It can be seen that the node activity distributions have heavy tails. In other words, a few network core nodes in MOSNs hold a large fraction of total contact frequencies. Therefore, these two traces satisfy the top-heavy property.

Now, let us look into more MOSN traces in [13–16], where all nodes in the network are classified into internal nodes and external nodes. Only internal nodes are analyzed, since contacts between external nodes are not collected. The results are shown in Table I. A remarkable common feature of these traces is the small network diameter. For Intel and Cambridge traces, the diameters of 1 imply clique structure. This is because wireless devices meet each other more or less during a long time period. Nodes are very easy to opportunistically meet each other, since transmission ranges of wireless devices are not small (compared to the scenarios). The key differences between nodes in MOSNs are their weighted degrees rather than degrees: a few nodes hold a large fraction of the total

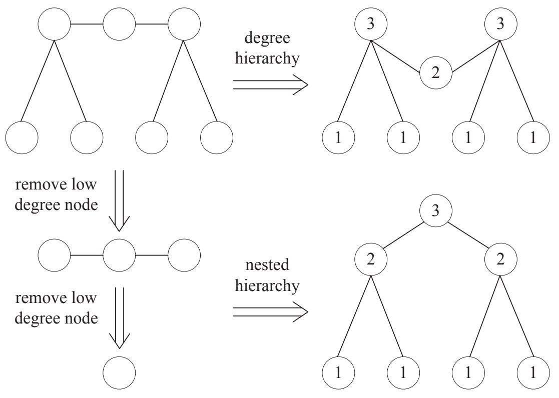


Fig. 4. An illustration of the subtle difference between the nested hierarchy and the degree (or weighted degree) hierarchy. A node labeled by a larger number has a higher position in the hierarchy.

contact frequencies. This kind of top-heavy property is shown in the second-to-last column of Table I.

C. Nested Hierarchies of MOSNs

At this time, it can be clearly seen that MOSNs have both nested core-periphery properties and top-heavy properties. Then, an interesting observation is that MOSNs have nested hierarchies. Note that the nested hierarchy is totally different from the degree or weighted degree hierarchies used in BubbleRap [7]. For a clear explanation, we define:

Definition 1: The nested hierarchy of a network defines the hierarchy of nodes in the network, through iteratively removing nodes with the lowest degrees (or weighted degrees). The nodes removed earlier have lower hierarchies, while the nodes removed later have higher hierarchies.

Definition 2: The degree (or weighted degree) hierarchy of a network defines the hierarchy of nodes in the network, according to the degree (or weighted degree) of the node. The nodes with smaller degrees (or weighted degrees) have lower hierarchies, while the nodes with larger degrees (or weighted degrees) have higher hierarchies.

To better show the subtle difference between these two hierarchies, an illustration is shown in Fig. 4. In this toy example, we consider the link weights to be identical for simplicity. The original network is shown in the top left corner of Fig. 4, while these two hierarchies are shown in the right side of Fig. 4. Then, the key observation and insight are described as follows. (1) The degree (or weighted degree) hierarchy has local minimums, i.e., the two nodes with the highest hierarchies are not connected. If we use the BubbleRap [7] scheme and iteratively forward the message to a encountered neighbor with a larger degree (or weighted degree) than the message holder, then local minimums are likely to be encountered and the message cannot be uploaded to the node with the largest degree (or weighted degree). (2) MOSNs have nested core-periphery structures. If we iteratively forward the message to an encountered neighbor that has a higher position in the nested hierarchy, then this message is likely to be forwarded to the node that has a highest position in the nested hierarchy. In other words, there are *much fewer local minimums* in the nested hierarchy, since MOSNs inherit nested core-periphery structures from social networks.

Therefore, the nested core-periphery structures of MOSNs provide conveniences for the message upload. However, a distributed pre-processing is needed to label the hierarchies of nodes. Since node activities are much more important than node connectivities for information dissemination in MOSNs, the nested hierarchy of the weighted degree version is adopted in our upload scheme. Then, the labeling scheme that determines the hierarchies of nodes is shown as follows:

- Effective weighted degree of a node is a summation of weighted degrees among all its unlabeled neighbors.
- A node labels itself, only when it has the lowest effective weighted degree among all its unlabeled neighbors. The label is set to be the largest label among its labeled neighbors plus one.

A larger label means that the corresponding node has a higher position in the nested hierarchy. Actually, this distributed labeling scheme is equivalent to the process of iteratively removing the nodes with lowest weighted degrees. A node that is labeled earlier has a lower hierarchy, while a node that is labeled later has a higher hierarchy. Then, we define:

Definition 3: When all nodes in an MOSN are labeled, a node is called a root node, if it has the largest label among all its neighbors.

As previously mentioned, MOSNs are likely to have very few root nodes. For further verification, we also look into the real MOSN traces [13–16] that are analyzed in Table I. The last column of Table I shows the number of root nodes in these traces. One amazing observation is that these traces have only one root node. In this sense, we have:

Theorem 1: In an MOSN with only one root node, the message can be uploaded from any node to the root, if the message holders iteratively forward the message to the relays that have larger labels than they do themselves.

Theorem 1 is very intuitive, which can be proved by a simple contradiction. We also propose strategies to cope with the possibility of local minimums (i.e., multiple root nodes) in Section V. Obviously, the nested hierarchy of MOSNs comes handy for optimizing the routings, especially for the upload phase. All nodes can find the root through iteratively forwarding the message to the relays that have larger labels than they do themselves. However, the root is not likely to know all the other nodes and corresponding efficient routing paths. Therefore, prior information is necessary for the message download from core nodes to the destination. In the next section, Bloom-filter-based routing hints are introduced to guide the download routing phase.

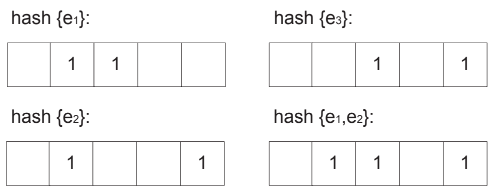


Fig. 5. A traditional Bloom filter ($m = 5$, and $k = 2$). If $\{e_1, e_2\}$ are added, a query for e_3 returns “in the set”, which is a false positive.

IV. Bloom-Filter-Based Routing Hints

In this section, we first present some preliminaries on Bloom filters, followed by the detailed routing hint construction to guide the download routing phase. Then, the routing hint update process is shown.

A. Preliminaries on Bloom Filters

The traditional Bloom filter [17] is a space-efficient probabilistic data structure, which is used, through the query operation, to check whether an element is a member of a set. The returned answer of “in the set” may be wrong, the probability of which is called false positive probability. Meanwhile, the answer of “not in the set” is definitely correct.

Assume there are a total of n elements (denoted as e_1 to e_n) inserted into the Bloom filter, where each element belongs to $\{1, 2, ..., N\}$ ($n \ll N$). A traditional Bloom filter employs a bit array of size m ($n < m \ll N$) to store these elements, as illustrated in Fig. 5. There are k different independent hash functions (denoted as h_1 to h_k), each of which maps the input set to one of the m array positions. Initially, all bits in the array are 0. To add an element e_i, the bits of array positions $h_1(e_i), ..., h_k(e_i)$ are set to be 1. To query an element e_j, k bits in the array positions of $h_1(e_j), ..., h_k(e_j)$ are checked. If any bits are 0, then “not in the set” is returned. Otherwise, it returns “in the set.” Since the k bit positions of e_j may be covered by some other elements, e_j may be incorrectly verified as existing, resulting in a false positive. The probability of false positives is described in [17] as follows:

$$\left[1 - (1 - \frac{1}{m})^{nk}\right]^k \approx (1 - e^{-kn/m})^k \tag{2}$$

The k that minimizes the probability of false positives is

$$k = \frac{m}{n} \ln 2 \tag{3}$$

Bloom filter is a tradeoff between the time complexity of a query and the space complexity of the storage. If we use a simple array to store these n elements, the time complexity of a query is $O(1)$, and the space complexity of the storage is $O(N)$. If we use a linked list, the time complexity is $O(n)$ and the space complexity is $O(n)$. Note that, for Bloom filters, the time complexity is $O(k)$ and the space complexity is $O(m)$, while generally we have $k \ll n < m \ll N$.

B. Routing Hints

Routing hints are essential for the download routing phase, since the destination may not be connected to the network

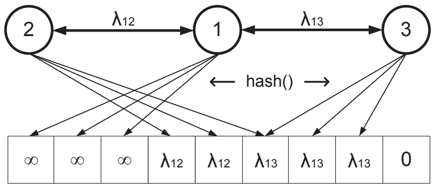


Fig. 6. The Bloom-filter-based routing hint of node 1.

core directly. In traditional IP networks, each node records the best forwarder for a specified destination (the Bellman-Ford algorithm in RIP) as the routing hints (or tables). However, the node storage capacity is extremely limited in MOSNs, where tracking all the connectivity information is not feasible. Therefore, Bloom-filter-based hints are introduced to reduce the hint storage demand at the cost of hint accuracy. *The hints locate within the MOSN nodes, while the sizes of hints are not the same* (described later in Theorem 2).

Without loss of generality, we assume that the total number of nodes in the MOSN is N, with node ID ranging from 1 to N. Then, the proposed routing hint is an extension of the Bloom filter, where the bit table is replaced by a weighted table to record the contact frequency. An example is shown in Fig. 6. Initially, each node stores the contact frequencies of its neighbors, from lower node ID to higher node ID. For example, node 2 is found as a neighbor of node 1, the corresponding table position of which is set to be λ_{12}. The same thing happens for recording node 3. However, the 6th table position has been taken by node 2, the value of which is then refreshed. Finally, the routing hint of node 1 stores itself, the corresponding table positions of which are set to be ∞. The routing hint also supports query operations, which returns the frequency of a queried node. A query operation looks into all the k array positions of the queried node, and returns the most "popular" value (the value that appears most frequently), if these positions are all non-zero values. For example, if node 2 is queried (for the routing hint of node 1), λ_{12} is returned, since it appears twice, and λ_{13} only appears once. If there is a table position of value 0 among the k positions, 0 is returned. For node i with Bloom filter T, the corresponding hint initialization and query operation ($Initialization_i$ and $Query_i$) are described in Algorithms 1 and 2 (for each node, how to determine m and k is described later).

The time complexity of the hint initialization is the same as the Bloom filter, i.e., $O(nk)$. However, the time complexity of the query operation is changed to be $O(k \log k)$, since it needs to sort the values of the k array positions to find the most "popular" one. Since k is generally a small value, $O(k \log k)$ is an acceptable cost. Another advantage of the proposed routing hint is that it supports the deletion operation, which is the inverse operation of the query operation (delete the most "popular" elements). The deletion operation provides a method to update the connectivity information, which is important for the MOSNs. In the next subsection, we will describe the routing hint update, as well as the principal to determine hint size for each node.

C. Periodic and Directional Routing Hint Update

Inspired by the study on the nested core-periphery network structure, we propose that the routing hint of a node is only

Algorithm 1 $Initialization_i$

```
Input: IDs and contact frequencies of the neighbors;
1: Initialize a weighted array $T$ of size $m$;
2: Initialize $k$ independent hash functions $h_1, ..., h_k$;
3: for each neighbor $j$ of node $i$ do
4:     for each hash function $h$ from $h_1$ to $h_k$ do
5:         Set $T[h(j)] = \lambda_{ij}$;
6: Set the corresponding positions of node $i$ to be $\infty$;
```

Algorithm 2 $Query_i$

```
Input:   A specified node ID $j$;
Output:  The contact frequency $\lambda_{ij}$;
1: Initialize a weighted array $R$ of size $k$;
2: for $s = 1$ to $k$ do
3:     Set $R[s] = T[h_s(j)]$;
4: if $R$ contains an element of $0$ then
5:     return $0$;
6: else
7:     return the most "popular" element in $R$;
```

Algorithm 3 $Update_i$

```
Input: Routing tables from all the neighbors of node $i$;
1: for each neighbor $j$ do
2:     if $label(j) < label(i)$ then
3:         for each possible node ID $e$ do
4:             if $1/Query_j(e) + 1/\lambda_{ij} < 1/Query_i(e)$ then
5:                 for each hash function $h$ from $h_1$ to $h_k$ do
6:                     Set $T[h(e)] = 1/(1/Query_j(e) + 1/\lambda_{ij})$;
```

needed to be sent to adjacent neighbors that have larger labels than that node (i.e., neighbors that have higher positions in the nested network hierarchy than that node). In other words, the routing hint update is directional, which is totally different from the Bellman-Ford algorithm in RIP. Directional updates result in a relatively small cost on the routing hint maintenance. Moreover, the structure of the directional routing hint update is *a DAG rather than a tree*, as shown in Fig. 7. A node sends its routing hint to multiple higher-level neighbors (i.e., its parents in the DAG). This kind of DAG structure will enable multiple paths in the download phase (discussed later in Section V).

In our scheme, *heterogeneous routing hint size* is employed to save the storage space, where the major periphery nodes record less information, and the minor core nodes record more information. Based on the received routing hints, the node updates its routing hint to record the estimated contact frequency (or *the expected shortest path delay*) with the other node. Note that the reciprocal of the contact frequency represents expected data delivery delay along the shortest path. An example is shown in Fig. 7. After receiving the routing hint from node 2 (which contains the information on node 3 and λ_{23}), node 1 updates its routing hint to set up routing information on node 3. Since the expected delay from node 1 to 3 is $(1/\lambda_{12}+1/\lambda_{23})$, node 1 set λ_{13} to be $1/(1/\lambda_{12}+1/\lambda_{23})$. The algorithm for the routing hint update of node i (denoted as $Update_i$) is presented in Algorithm 3. Note that the hint update is a periodic process, which is similar to the periodic exchange routing tables in IP networks.

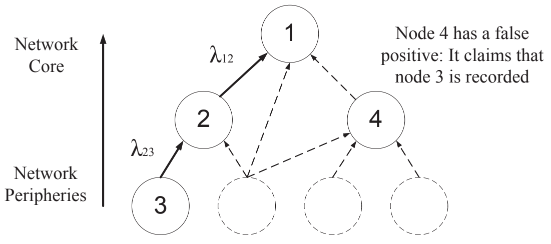


Fig. 7. An example of the routing table updates, where the arrow represents the information flow of the routing tables. Node 3 sends its routing table to node 2. After updating, node 2 sends its routing table to node 1. Note that this is a DAG structure instead of a tree.

Now, let us determine a reasonable routing hint size for a node with degree d. Then, we have:

Theorem 2: According to Algorithm 3, the expected number of inserted elements in the routing hint of a node with degree d (denoted as n_d), satisfies $n_d \leq d(\alpha-1)^{d-2}$, if (1) the corresponding network has a power-law degree distribution, and (2) the label of a node is proportional to its degree.

The proof of Theorem 2 is shown in Appendix A. In this theorem, α is a parameter of the power-law degree distribution, the assumption of which indicates the top-heavy property (see Appendix B). Although the assumptions made in Theorem 2 are kind of restrictive, it is still an effective coarse estimation of the routing hint size m. For practical usage, the routing hint size of a node with degree d can be set as constant times as large as $d(\alpha-1)^{d-2}$. For example, $m = 10 * d(\alpha-1)^{d-2}$. Meanwhile, the number of desired hash functions, k, can be determined by Eq. 3. At this time, the probability of false positives is $0.6185^{m/n}$. Again, note that nodes with different degrees keep different sizes of routing hints, while the network core nodes keep more information, since they usually have higher degrees. Obviously, network core nodes are generally more active and more powerful, and thus they are able to have larger sizes of routing hints. Although this is unfair for core nodes, "hotspot" effects are inevitable in MOSN routings. This is because network core nodes are bottlenecks for high-performance routing algorithms. In our future work, we will further extend Theorem 2 to *determine a proper number of copies* for the download phase to both effectively resist false positives and take advantages of multiple routing paths.

V. Up-and-Down Routing Protocol

In this section, we describe the whole up-and-down routing protocol, which includes an upload routing phase and a download routing phase. The message upload phase is based on the nested hierarchies of MOSNs, as shown in Theorem 1. Then, the message download phase is implemented through space-efficient Bloom-filter-based routing hints, the sizes of which can be bounded by Theorem 2. These two phases are respectively described in the following two subsections.

A. Upload Phase

In the previous discussions, Theorem 1 points out that the source can upload its message to the unique root node, through iteratively forwarding the message to the inter-meeting relays that have higher positions (i.e., nodes with larger labels) in the nested network hierarchy than the message

Algorithm 4 Upload Phase of Node i

```
Input: Routing tables of node $i$ and its neighbors;
1: for each neighbor $j$ do
2:     if $Query_j(destiantion) > 0$ then
3:         Upload phase terminates, download phase starts;
4: for the inter-meeting neighbor $j$ do
5:     if $label(i) < label(j)$ then
6:         Forward the message to node $j$;
```

Algorithm 5 Download Phase of Node i with c Copies

```
Input: Routing tables of node $i$ and its neighbors;
1: for the inter-meeting neighbor $j$ do
2:     $a = Query_i(destiantion)$;
3:     $b = Query_j(destiantion)$;
4:     if $label(i) > label(j)$ then
5:         Hand over $\lceil b \times c/(a+b) \rceil$ copies to node $j$;
6:         Update $c$ to be the number of the remaining copies;
```

holder. However, it is not necessary for the message to be uploaded to the root, since there may be some shortcuts. Therefore, the upload termination condition is that the message is forwarded to a node, where the destination-related hint (i.e., $Query(destination)$>0) is available among itself and its neighbors, rather than achieving the root. Note that the routing hint of the root contains all the nodes in the network. There are also some alternative termination conditions, e.g., upload until achieving several qualified neighbors. In the upload phase, we use only one message copy, since it achieves competitive delivery delay with a low forwarding cost. The insight is that we optimize the routing performance by utilizing the prior knowledge on the network structure. As a summary, the upload phase is described in Algorithm 4.

Another potential problem for the upload phase is that, multiple root nodes may exist in large MOSNs. A more intuitive counterexample is that the two nodes with the highest and second highest labels may not connect with each other, due to the node mobility limitations in large-scale MOSNs. Fortunately, there are several methods for solving the existence of multiple root nodes, i.e., local minimums. These methods are shown as follows. (1) A costly but effective method is to use broadcasting for the root nodes to discover each other and then set up *virtual connections*. The virtual connections can be implemented as a pre-determined routing path. (2) Instead of the virtual connections, the second method is the logical link removal. If we logically disable part of the existing links, some root nodes can be removed (i.e., they are no longer root nodes due to the disappearance of partial links). (3) The third method is using probabilistic uploads, where the message can jump out of a root with a carefully designed probability. This idea is borrowed from the simulated annealing algorithm [18].

Since the multiple roots problem is potential, but not observed in current real traces (i.e., real trace simulations are not available), we do not further discuss it in this paper. The notable point is that the MOSNs naturally have nested core-periphery structures. Therefore, the nested hierarchies will bring much less local minimums than will the degree (or weighted degree) hierarchies used in BubbleRap [7].

B. Download Phase

Whenever an uploading message holder finds qualified neighbors, the upload phase switches to the download phase. In the upload phase, we only use one message copy to save the data forwarding cost, however, we use multiple copies in the download routing phase. There are two reasons for the usage of multiple copies as follows:

- The first reason is that the Bloom-filter-based routing hints have false positives (i.e., multiple nodes claim the destination as their descendant). Basically, *the hint storage is reduced at the cost of the routing hint accuracy*, which will cause a lower delivery ratio. Therefore, multiple copies are used to mitigate negative effects (in terms of data delivery ratio) from the inaccurate routing hints. As shown in Fig. 7, suppose node 1 wants to download a message to node 3 through two copies. Then, one copy is respectively passed to node 2 and 4, since they both claim that they have information on node 3 within their routing hints. However, the truth is that node 4 has a false positive and thus cannot deliver its copy to node 3.
- The second reason is that using multiple copies can effectively reduce the download delay. Although the routing hint within a relay only records the shortest path to the destination, the opportunistic nature can still accelerate the download phase. When the message holder opportunistically meets a neighbor with a secondary path (note that a DAG instead of a tree is maintained), partial copies will be passed to that neighbor to reduce download delay. As mentioned in Fig. 7, a DAG is constructed when each node claims more than one higher-level neighbor as its parents.

The node that terminates the upload phase would replicate the message for the download phase. In the download phase, the message copies are only allocated to *relays that have smaller labels than the message holder*. Moreover, the number of message copies allocated to the two nodes are proportional to their estimated contact frequencies (i.e., reciprocal of the expected delay) with the destination. This information is stored in the routing hints and can be accessed by the query operation. For example, if node i holds c copies, and node j contacts node i, then the number of copies allocated to node j is

$$\left\lceil \frac{Query_j(destination)}{Query_i(destination) + Query_j(destination)} \times c \right\rceil \quad (4)$$

The insight behind this copy allocation scheme is very simple: the node that has a lower expected shortest path delay is allocated with more message copies. A formal description of the download phase is provided in Algorithm 5. In the next section, we will evaluate the up-and-down routing scheme through extensive simulations.

VI. Evaluation

In this section, extensive simulations are conducted to evaluate the proposed routing protocol. After presenting the traces and the settings, we show the algorithms and metrics for comparison. Finally, the evaluation results are shown from different perspectives to provide insightful conclusions.

A. Traces and Settings

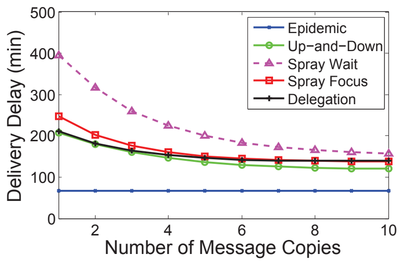
In our simulations, two traces are used. One is a real trace, the Sigcomm [16], which is one of the largest real traces (only internal nodes employed). The details of the Sigcomm trace are shown in Table I. The reason why we choose only one real trace is that the network structures are similar for different real traces. Since the real traces are too small-scale to extract α (the power-law parameter in Theorem 2, as to set the sizes of routing hints), we set $\alpha = 2.5$ as an estimation for the Sigcomm trace. The other trace is a synthetic trace produced by the preferential attachment model (100 nodes with average degree 10 in the Barabási-Albert model with $\alpha = 2.1$). More details are attached in Appendix B. In the synthetic trace, the weight of each link follows a uniform distribution in the interval $[0, 0.1]$. To guarantee connectivity, one random node in each small component is assigned a link to the node with the largest degree in the giant component. This synthetic trace will bring a microcosm of large-scale MOSNs.

In our simulations, the unit of the edge weight is determined as the number of contacts per minute. We use 500 minutes as the data delivery deadline. If the destination is not achieved before the deadline, then the data delivery is viewed as failed, and the deadline is regarded as the delivery delay. For the up-and-down routing protocol, we use the routing hints with size $m = 10 * d(\alpha - 1)^{d-2}$ according to Theorem 2. Here, the constant 10 is denoted as the *robustness ratio*, as to observe the influence of the false positives of the routing hints. A higher robustness ratio means the routing hints are more accurate (and also indicates higher storage demands), and thus are supposed to bring better performance. The influence of the robustness ratio on the routing performance is also observed and analyzed in our simulations.

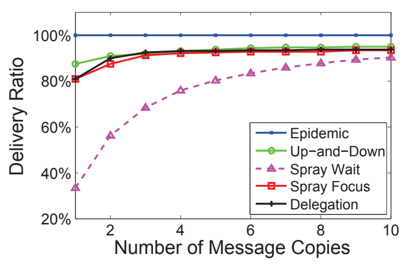
B. Algorithms and Metrics in Comparison

We assign the following four algorithms for comparison: (1) Epidemic, where the nodes continuously replicate and transmit messages to newly discovered nodes that do not already possess a copy. Epidemic represents the minimum data delivery delay (and the highest cost) of all routing algorithms; (2) (Binary) Spray and Wait [8], where the data delivery is composed of a spray phase and a wait phase; during the spray phase, the message holders share their copies with encountered nodes that have no copies. When a message holder has only one copy left, it enters the wait phase, and waits for meeting the destination; (3) (Binary) Spray and Focus, where the wait phase in the former algorithm is replaced by a single-copy utility routing scheme (forward the copy if the relay has a higher contact frequency than the destination); (4) Delegation Forwarding, which is modified to use a limited number of copies. Instead of holding a copy for each message transmitter, the numbers of copies allocated to the message holder and relay are proportional to their contact frequencies with the destination (similar to Eq. 4). In addition, the BubbleRap [7] is not used for comparison, since it requires an additional community information.

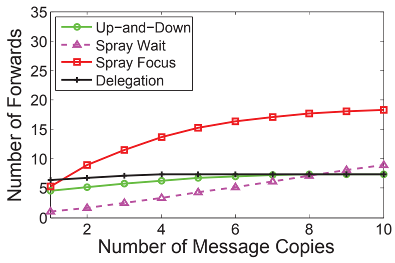
Three classical metrics are used to measure the quality of the up-and-down routing, including data delivery delay, ratio, and cost (the number of forwards when achieving the destination). For further analyses, three additional metrics are



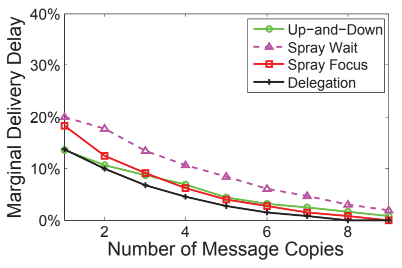
(a) Data Delivery Delay



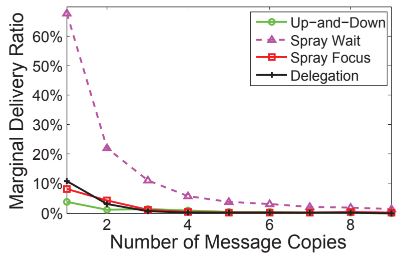
(b) Data Delivery Ratio



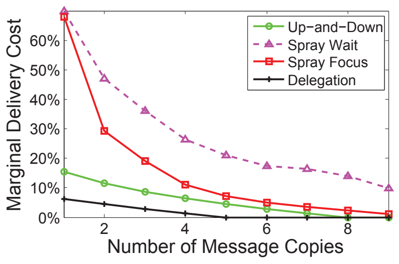
(c) Number of Forwards



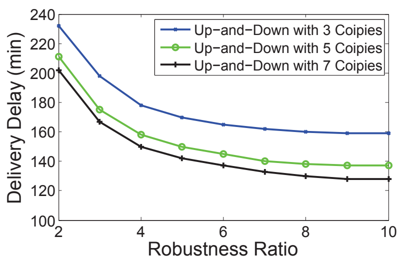
(d) Marginal Delivery Delay



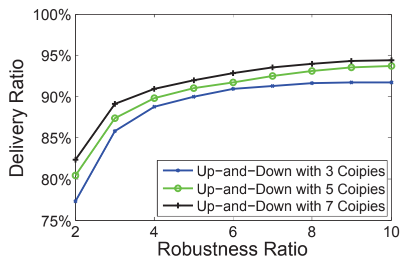
(e) Marginal Delivery Ratio



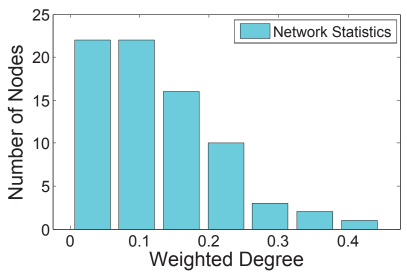
(f) Marginal Delivery Cost



(g) Robustness of Delivery Delay



(h) Robustness of Delivery Ratio



(i) Network Statistics

Fig. 8. Simulation results of the Sigcomm trace.

proposed: the marginal delivery delay, the marginal delivery ratio, and the marginal delivery cost. These metrics measure the percentages of reduced delivery delay, increased delivery ratio and cost brought *by using one more copy* in the download routing phase. Through these metrics, we can determine the appropriate number of copies to be used in our routing scheme.

C. Evaluation Results

The simulation results for the Sigcomm trace are shown in Fig. 8. It can been seen that up-and-down routing outperforms the other routing schemes (except for Epidemic) in the view of both data delivery delay and ratio. In the view of delivery cost (i.e., number of forwards), the up-and-down protocol is also good (only larger than the Spray and Wait). Note that the reason for a low cost if the Spray and Wait is that it has a low delivery ratio, since all the copies are waiting for the destination rather than being forwarded to better relays. Figures 8(d) to 8(f) show the marginal delivery delay, ratio and cost. Obviously, the number of copies follows the law of diminishing returns (decreased marginal delivery delay, ratio, and cost). We found that 4 copies are enough for the Sigcomm trace. Figures 8(g) and 8(h) show the influence of robustness ratio on the data delivery. When this ratio is less than 2, the routing scheme almost fails due to the inaccurate routing hints. Meanwhile, a robustness ratio of 10 is good enough to support the up-and-down routing scheme. Therefore, Theorem 2 gives out an effective estimation, in terms of the simulations. Finally, note that the weighted degree distribution of the Sigcomm trace is shown in Fig. 8(i).

The simulation results for the synthetic trace are presented in Fig. 9. Since routings in this trace usually need more than two hops, the delegation forwarding does not work, considering that all the neighbors of the source may not recognize the destination. In this situation, the proposed up-and-down routing protocol performs much better than the traditional schemes, in terms of the delivery delay and ratio in Figures 9(a) to 9(b). The cost of the proposed protocol is a little higher than the other schemes as a tradeoff to obtain much better delivery delay and ratio. In the view of marginal delivery delay and ratio, almost the same results are obtained as the Sigcomm trace. However, the delivery costs no longer have diminishing return for all algorithms. Finally, we also obtain similar results on the robustness ratio (the higher ratio the better), as shown in Figures 9(g) and 9(h).

The outstanding performance of the proposed up-and-down protocol has been verified by the evaluation results. Based on the network structure, the message is uploaded to the root

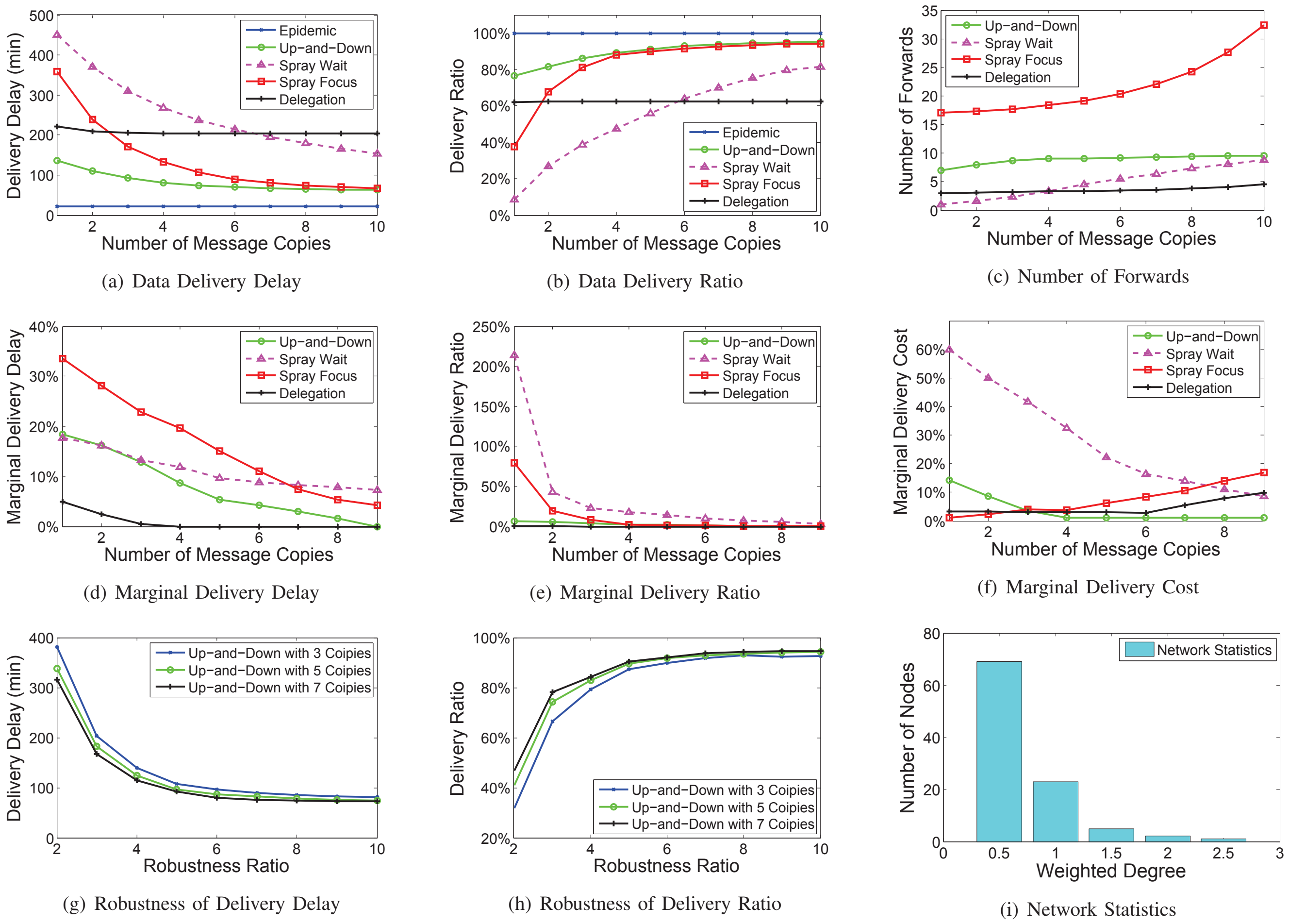


Fig. 9. Simulation results of the synthetic trace.

efficiently. Based on routing hints, the message download is also efficient. The proposed routing scheme achieves good performance in data delivery delay, ratio, and cost, as the tradeoff of acceptable storage usage for the routing hints.

VII. Conclusion

In this paper, we propose an up-and-down routing protocol, which has an upload phase and a download phase. Firstly, we verify that the nested core-periphery property and the top-heavy property both exist in MOSNs. The nested network hierarchy enables a message to be uploaded from the source to the network core, through iteratively forwarding the message to a relay that has a higher position in the nested network hierarchy. Then, space-efficient routing hints are proposed based on the Bloom filters, which provide guidance for the message download. Multiple copies are used in the download routing phase to resist inaccurate routing hints, as well as reduce the download delay. Through using the network structure and a few routing hints as prior information, subtle balances between the data delivery delay, ratio, and cost are achieved by our proposed approach. Finally, the simulation results show that good data delivery delay and ratio are achieved with a low-cost prior information collection process and a relatively low forwarding cost.

References

[1] A.-K. Pietilänen and C. Diot, "Dissemination in opportunistic social networks: the role of temporal communities," in *Proc. of MobiHoc 2012*, pp. 165–174.

[2] N. Jabeur, S. Zeadally, and B. Sayed, "Mobile social networking applications," *Commun. ACM*, vol. 56, no. 3, pp. 71–79, 2013.

[3] B. Han, P. Hui, V. S. A. Kumar, M. V. Marathe, J. Shao, and A. Srinivasan, "Mobile data offloading through opportunistic communications and social participation," *IEEE Trans. Mob. Comput.*, pp. 821–834, 2012.

[4] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proc. of WWW 2010*, pp. 631–640.

[5] M. E. J. Newman, "Power laws, pareto distributions and zipf's law," *Contemporary Physics*, vol. 46, pp. 323–351, 2005.

[6] S. team, "Contact networks in a primary school," Downloaded from http://wiki.gephi.org/index.php/Datasets, 2011.

[7] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: social-based forwarding in delay tolerant networks," in *Proc. of MobiHoc 2008*, pp. 241–250.

[8] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proc. of WDTN 2005*, pp. 252–259.

[9] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot, "Delegation forwarding," in *Proc. of MobiHoc 2008*, pp. 251–260.

[10] C. Liu and J. Wu, "Scalable routing in delay tolerant networks,"

in *Proc. of MobiHoc 2007*, pp. 51–60.
[11] Y. Tao and X.-f. Wang, "Adaptive clustering hierarchy routing for delay tolerant network," *Journal of Central South University*, vol. 19, pp. 1577–1582, 2012.
[12] A. Balasubramanian, B. Levine, and A. Venkataramani, "Replication routing in dtns: A resource allocation approach," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 596–609, 2010.
[13] N. Eagle, A. S. Pentland, and D. Lazer, "Inferring friendship network structure by using mobile phone data," *PNAS*, vol. 106, no. 36, pp. 15 274–15 278, 2009.
[14] G. Bigwood, D. Rehunathan, M. Bateman, T. Henderson, and S. Bhatti, "CRAWDAD trace set st_andrews/sassy/mobile (v. 2011-06-03)," Downloaded from http://crawdad.cs.dartmouth.edu/st_andrews/sassy/mobile, Jun. 2011.
[15] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD data set cambridge/haggle (v. 2009-05-29)," Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle, May 2009.
[16] A.-K. Pietilainen and C. Diot, "CRAWDAD data set thlab/ sigcomm2009 (v. 2012-07-15)," Downloaded from http://crawdad.cs.dartmouth.edu/thlab/sigcomm2009, Jul. 2012.
[17] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: A survey," in *Internet Mathematics*, vol. 1, no. 4, 2002, pp. 636–646.
[18] H. Zheng, "An improved niche genetic algorithm based on simulated annealing: Sanga," in *Proc. of ICCP 2011*, pp. 1–5.
[19] M. Newman, *Networks: An Introduction*. New York, NY, USA: Oxford University Press, Inc., 2010.
[20] A.-L. Barabási, "Scale-Free Networks: A Decade and Beyond," *Science*, vol. 325, no. 5939, pp. 412–413, Jul. 2009.
[21] H. Jeong, Z. Neda, and A. L. Barabási, "Measuring preferential attachment in evolving networks," *Europhysics Letters*, vol. 61, no. 4, pp. 567–572, 2007.

Appendix

A. Proof of Theorem 2

Now, let us assume that the network has a power-law degree distribution, i.e., the fraction (denoted as P_d) of nodes with degree d is proportional to $d^{-\alpha}$:

$$P_d = (\alpha - 1)d^{-\alpha} \quad where \quad d \in \{1, 2, 3, ..., \infty\} \tag{5}$$

Here, α is the power-law distribution parameter that usually satisfies $\alpha \in [2, 3]$ in real networks [19]. Another point is that, in Section III, we have assumed the MOSNs to be symmetric (i.e., in-degree equals out-degree), and here we use degree to denote both in-degree and out-degree. Note that this assumption is reasonable, since MOSNs have the top-heavy property, which can be usually inferred by the power-law degree distribution [5]. More materials can be found in [19].

According to Eq. 5, the fraction W_d of free ends of edges provided by nodes with degree d is

$$W_d = \frac{dP_d}{\int_1^\infty iP_i \mathrm{d}i} = \frac{(\alpha - 1)d^{1-\alpha}}{(\alpha - 1)/(\alpha - 2)} = (\alpha - 2)d^{1-\alpha} \tag{6}$$

Note that the fraction of ends of edges, which are attached to a node with d (given by Eq. 6) is $(\alpha - 2)d^{1-\alpha}$. Assuming the label of a node is proportional to its degree, then n_d ($d \geq 2$) can be calculated by

$$n_d = d\sum_{i=2}^{d-1}(\alpha - 2)i^{1-\alpha}n_i = d(\alpha - 2)\sum_{i=2}^{d-1}\frac{n_i}{i^{\alpha-1}} \tag{7}$$

The meaning of Eq. 7 is shown as follows. (1) For each edge of the node with a degree d, the probability of connecting to a lower degree node with a degree i is $(\alpha - 2)i^{1-\alpha}$. (2) The node with a degree i brings n_i elements to the routing hint of the node with a degree d. (3) Here, the requirement of $d \geq 2$ is that nodes with degree one do not need to store the routing hint. This is because the nodes with degree one are at the bottom of the network hierarchy.

Since the expression of n_{d-1} is similar to the expression of n_d in Eq. 7, we have the following equation:

$$\frac{n_d}{d(\alpha - 2)} - \frac{n_{d-1}}{(d-1)(\alpha - 2)} = \frac{n_{d-1}}{(d-1)^{\alpha-1}} \tag{8}$$

Eq. 8 can be rewritten as follows:

$$n_d = \Big[\frac{d}{d-1} + \frac{d(\alpha - 2)}{(d-1)^{\alpha-1}}\Big]n_{d-1} \tag{9}$$

Since $\alpha \in [2, 3]$, we have $(d-1)^{\alpha-1} > (d-1)$. Therefore, Eq. 9 can be simplified as follows:

$$n_d \leq \Big[\frac{d}{d-1} + \frac{d(\alpha - 2)}{d-1}\Big]n_{d-1} = \frac{d(\alpha - 1)}{d-1}n_{d-1} \tag{10}$$

Note that $n_2 \leq 2$, since a node with degree two has at most two records. If we do the recursion in Eq. 10, then we get:

$$n_d \leq d(\alpha - 1)^{d-2} \tag{11}$$

Now, Eq. 11 shows the expected number of inserted elements in the routing hint of a node with degree d.

B. The Preferential Attachment Model

In the past decades, researchers tacitly assumed components of complex network systems (such as society and the Internet) to be randomly wired together, which is impractical for real networks. In recent years, an avalanche of research has shown that many real networks can be abstracted as scale-free network architectures [20]. The key feature of these networks is the power-law degree distribution [19], as analyzed in Appendix A.

Currently, the Barabási-Albert model is one of the most acknowledged models for generating the scale-free networks [21]. In this model, nodes are iteratively added one by one to a growing network, and each new node connects to a suitably chosen set of previously existing nodes. The probability that the new node is connected to a previous node i is proportional to node i's degree plus a constant. This constant is used to tune the corresponding power-law distribution parameter α. Obviously, the existing node with a larger degree is more intended to connect to the new nodes (the rich get richer, and the poor get poorer), and thus a large number of edges are connected to a few nodes with the highest degrees. This phenomenon has been verified in the real networks [5]:

Proposition 1: Let W denote the fraction of ends of edges, which are attached to a fraction P of the highest-degree nodes. In scale-free networks, W and P satisfy $W = P^{(\alpha-2)/(\alpha-1)}$.

The proof is shown in [19]. Since generally $\alpha \in [2, 3]$ (see [19]), $\frac{\alpha-2}{\alpha-1}$ is in the range of $[0, 0.5]$, i.e., $W > P$. This implies that a few network core nodes hold a large fraction of links, which is also found as the top-heavy property in MOSNs.