How Much *off-center* Are Centrality Metrics for Routing in Opportunistic Networks

Pavlos Nikolopoulos Therapon Papadimitriou Panagiotis Pantazopoulos
Merkourios Karaliopoulos Ioannis Stavrakakis
Department of Informatics and Telecommunications - National & Kapodistrian University of Athens
Ilissia, 157 84 Athens, Greece
{p.nikolopoulos, theraponpap, ppantaz, mkaralio, ioannis}@di.uoa.gr

ABSTRACT

The exploitation of social context for routing data in opportunistic networks is a relatively recent trend. Node centrality metrics, such as the betweenness centrality, quantify the relaying utility of network nodes and inform routing decisions, resulting in better performance than more naive routing approaches. Nevertheless, centrality-based routing is far from optimal for three main reasons: a) routing decisions are greedy and message *destination-agnostic*; b) its performance is highly sensitive to the *contact graph* over which the node centrality values are computed; c) the global network centrality values have for practical reasons to be approximated by their *egocentric* counterparts. Our paper experimentally assesses the impact of these three factors on the efficacy of centrality-based routing. Five centrality-based routing variants are compared with each other and against two schemes representing extreme instances of DTN routing complexity: the simple probabilistic forwarding protocol and an ideal scheme with perfect knowledge of future contacts that computes optimal message space-time paths over a novel graph construct with contacts as vertices and time-weighted edges. The results of this comparison are not always inline with intuition and indicate inherent weaknesses of centrality-based routing.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Store and forward networks, Wireless communication*

General Terms

Algorithms, Design, Performance

Keywords

Delay Tolerant Networks, Centrality, Space-time Graphs

CHANTS'11, September 23, 2011, Las Vegas, Nevada, USA.

1. INTRODUCTION

The exploitation of social context for improving the efficiency of routing in opportunistic networks is a relatively recent trend. Initial approaches to the DTN routing problem have essentially been variants of controlled flooding across the network. These schemes reduce the cost of pure epidemic dissemination by setting upper bounds on the message replication at the message source and/or relay nodes (*e.g.*, [15] and [7]). More informed decisions are made by forwarding schemes that try to assess the relaying significance (*utility*) of encountered nodes. *Node utility-based* forwarding accounts for the frequency of encounters with the destination node [11], or more general social context such as preferably visited places or interests/hobbies common to the candidate relay and the destination node [1].

More recently, social information has been introduced more formally into the node relaying utility functions through direct reuse of concepts and metrics from social network analysis (SNA). Examples of this approach are the SimBetTS and BubbleRap protocols. In both cases, the SNA metrics are computed over contact graphs that effectively aggregate the sequence of node encounters over certain time windows. In SimBetTS [5] the nodes' utilities are sums of their centrality, similarity, and tie strength values, the latter reflecting the frequency, duration, and recency of the contacts with other nodes. Whereas, BubbleRap [9] explicitly assumes the existence of nodes' communities and manipulates the centrality of encountered nodes, both within their communities and globally across the whole network, to route messages within and across these communities, respectively.

Both SimBetTS and Bubble Rap have reported enhanced performance over the controlled flooding and original utility-based forwarding approaches and identified centrality as the metric with the dominant impact on routing even when it is combined with other social metrics [5] [9]. Nevertheless, there are three main concerns regarding the centrality-based routing approaches. First of all, node centrality values are destination-agnostic; namely, the node relaying utilities are averages computed across all node pairs in the network (or community). Secondly, SNA metrics are computed over graphs. The derivation of these graphs out of the sequence (history) of contacts is not a straightforward process. Hossman *et al.* in [8] experimentally show that the performance of socioaware DTN routing protocols is highly sensitive to the way the social contact graph is constructed. They argue in favor of aggregating the DTN contacts targeting a fixed *edge* density rather than over fixed *time windows* and propose a machine-learning algorithm for the online determination of

the optimal aggregation density. Thirdly, the original centrality metrics need to be approximated by egocentrically computed centrality variants, which, in principle, can offer only limited views of the node's utility in the network.

Our work looks closer into these three issues and their impact on centrality-based DTN routing. We put one of the main SNA metrics, betweenness centrality (BC), under the microscope and study how the type of contact graph (unweighted *vs.* weighted) and the way the metric is computed (sociocentric *vs.* egocentric) affect the accuracy of BC-based forwarding decisions. Moreover, we assess how BC compares with its destination-aware variant, Conditional BC, which maintains per-destination centrality values for each node. We are interested in inherent weaknesses of centrality-based routing that pose hard limits to the performance of socioaware DTN routing rather than imperfections of particular protocol implementations: How close-to-optimal can routing decisions based on centrality metrics be? How do different alternatives for *computing* node centrality affect routing performance?

We elaborate on the alternatives for computing centrality in Section 2 and detail our experimentation methodology in Section 3. Five centrality-based routing variants are compared with each other and against two schemes representing extreme instances of DTN routing : the simple probabilistic forwarding protocol and an ideal scheme that computes optimal message paths with perfect knowledge of future contacts. Next, we introduce and analyze a graph representation of the contact traces that allows computing 'shortest' paths via direct application of standard shortest-path algorithms. We present our evaluation results in Section 5 and summarize our findings in Section 6.

2. IMPLEMENTATION ALTERNATIVES FOR CENTRALITY-BASED ROUTING

We iterate three pairs of alternatives related to the implementation of centrality-based routing.

Destination-aware *vs.* destination-unaware metrics. Ideally, when nodes decide to forward (or not) a message to an encountered node, they should take into account the relaying value of the node with respect to the message destination. On the other hand, centrality metrics can be, and often are in literature [5] [9], aggregate measures of nodes' significance, averaged over all network node pairs. Apparently, a node with high(low) centrality value may well be a poor(excellent) relaying choice for certain destination nodes.

Unweighted *vs.* weighted contact graphs. The practice is to compute node centrality metrics on the *contact graph*, which is extracted out of the contacts' sequence. In almost all studies so far, the contact sequence is transformed to a static graph $G = (V, E)$, where V is the set of network nodes and edges $e \in E$ join node pairs that have experienced $C_e > thr_e$ encounters within a fixed-duration time window T; popular values of $\{thr_e, T\}$ are $\{1, 6hrs\}$, see *e.g.*, [9]. Predictably, the performance of the centrality-based routing protocols is highly sensitive to both thr_e and T [8]. Furthermore, a given contact sequence can be also transformed to a *weighted* graph, with weights reflecting the frequency of encounters between node pairs. Computation of node centrality values over this weighted graph results in yet another set of node centrality values for the same contact sequence.

Egocentric *vs.* sociocentric computation of centrality metrics. The computation of centrality metrics usually requires global information about the network topology. Even when the approximation error induced by the transformation of contact sequences to static graphs is acceptable, individual DTN nodes cannot always be aware of the full social graph resulting from the encounters of all network nodes. Therefore, realistic protocol implementations such as [5] and [9] rely on *egocentric* approximations of the original sociocentric metrics, computed across the nodes' *ego networks* [12],[6]. However, there is little evidence regarding how well the original sociocentric metrics correlate with their egocentric counterparts in DTN contact graphs –or even when is this really a requirement and when not.

Our aim is to assess the informative value of node centrality metrics when they are used as routing criteria in opportunistic settings. These metrics are reported to have the dominant impact on the routing protocol performance even when they are jointly considered with other SNA metrics in routing decisions [9] [5]. Herein, we consider the most popular centrality metric in the context of DTN routing, betweenness centrality (BC), and question how the routing performance is affected by the three computational alternatives listed in Section 2. We argue that they pose the hard constraints on the achievable routing performance rather than the imperfections of particular protocol implementations or inaccuracies related to protocol parameter estimation.

Table 1: Characteristics of experimentation datasets

	Intel	Cambridge	Infocom05	Content	Infocom06
Device type	iMote	iMote	iMote	iMote	iMote
Duration(days)	6	6	4	24	4
Scan time(sec)	5-10	5-10	5-10	5-10	5-10
Granularity(sec)	120	120	120	120-600	120
Mobile Devices	8	12	41	36	78
Stationary Dev.	1	0	0	18	20
External Dev.	119	211	233	11368	4421
Average internal contacts/pair/day	9.09	12.09	8.60	0.66	9.03
# of Contacts	2766	6732	28216	41330	227657

3. EXPERIMENTATION METHODOLOGY

Our evaluation uses real traces of pairwise node encounters. We conduct experiments with five well-known experimental traces, gathered over the last five years in the context of the Haggle Project [4]. They include Bluetooth sightings by users carrying iMotes during the experiments. Each Bluetooth sighting is assumed to be a contact whereby nodes can exchange information. In Table 1, scan time is the time needed by iMotes to perform a complete scan for Bluetooth devices and takes approximately 5 to 10 seconds; whereas time granularity represents the idle time between two consecutive scans and turns out to be critical for the measurements' accuracy. Both contacts between iMotes ('internal' contacts) and other Bluetooth-enabled devices ('external' contacts) in the vicinity of iMote carriers are logged. Herein we analyze the internal contacts; these represent data transfer opportunities among the experiment participants, assuming that they are all equipped with always-on devices. The experimental settings are detailed in [4].

We generate message triplets $msg(s, d, t)$, where the message source s, destination d, and generation time t are randomly chosen, and emulate their paths over the traces. As the trace is replayed (sequentially read), network nodes compute online their BC values and make forwarding decisions for each message. We consider five possible ways to compute betweenness centrality values out of the history of contacts.

3.1 Emulation of centrality-based routing over the traces

The first processing step when implementing centrality-driven routing is the transformation of the encounters' history to a graph, over which the node centrality values are computed. This is an aggregation process that effectively compresses the time dimension. In this work, we produce two different static graph representations for each trace: one unweighted with $thr_e = 1$ and one weighted, where the edge weight equals the inverse of their encounters' count.

The generated $msg(s, d, t)$ messages are routed to their destination through successive 'greedy' forwarding decisions. More specifically, if at time t_j the message is with node u, then u will forward the message to another node, say k, upon its next encounter with it, as long as its betweenness centrality value $BC(k)$ is higher than its own BC value, $BC(u)$[1]. The BC values may be either the socio- or their ego-centric counterparts and are computed taking into account the full set of contacts within the time window $[t_j - T, t_j]$, where T is the contact aggregation interval. We assume that nodes avail perfect information about the history of encounters in the network when computing the global (sociocentric) BC values. Hence, any routing performance penalty is due to the (lack of) informative power of the metric rather than information unavailability. Besides the original BC metric, we experiment with its destination-aware variant, called Conditional Betweenness Centrality (CBC). We have introduced CBC [14] to capture the centrality of a random node with respect to a specific destination node d. It is defined as

$$CBC(u; d) = \sum_{s \in V, u \neq d} \frac{\sigma_{sd}(u)}{\sigma_{sd}} \quad (1)$$

with $\sigma_{sd}(s) = 0$. The summation is over all node pairs (x, d), $\forall x \in V$, destined at node d rather than all possible pairs, as in $BC(u)$.

Overall, the message routing process is emulated five times over a given trace of encounters. We use the abbreviation terms $soc(C)BC_{uw}$, $egoBC_{uw}$, $socBC_w$, $egoBC_w$, for routing based on socio/egocentric (C)BC values estimated on unweighted($_{uw}$)/weighted($_w$) graphs, respectively. We compute the message delivery delay and number of forwarding hops for all $msg(s, d, t)$ triplets and compare them with their counterparts under two reference schemes: a probabilistic forwarding scheme PF_p, whereby a node forwards a carried message with probability p upon every encounter, and the optimal(*opt*) scheme, described below. As centrality-driven routing is a greedy approach to the message path optimization problem, there is a non-negligible probability that a message gets trapped at a highly central node that never encounters the destination. Therefore, for each centrality-based routing variant, we also report the percentage of delivered messages out of the total messages that can be delivered by the optimal scheme (*deliverable* messages).

4. COMPUTATION OF OPTIMAL SPACE-TIME PATHS

The fastest possible paths to the message destination are computed directly out of the sequence of nodes' encounters; they correspond to the foremost journeys, as defined in the context of evolving graphs in [2]. To actually discover and make use of these paths, a node should have perfect knowledge about the exact sequence of future contacts between network nodes. This is clearly impossible except for a very few special cases, where nodes follow fully deterministic, often periodic, trajectories [10]. On the contrary, in general DTN instances the foremost journey can be obtained with application of the epidemic forwarding protocol [16] in the absence of buffer and link transmission rate limitations.

[1]This corresponds to the RANK algorithm in [9].

Contact id	Involved nodes		Contact time	Add. Fields
C_1	n_1	n_2	t_1	…
C_2	n_3	n_4	t_2	…
C_3	n_4	n_5	t_3	…
C_4	n_2	n_5	t_4	…
C_5	n_5	n_3	t_5	…
C_6	n_3	n_2	t_6	…
C_7	n_4	n_6	t_7	…
C_8	n_2	n_6	t_8	…

Figure 1: Sequence of contact entries C_i ($t_i > t_j$ for $i > j$). Bold entries denote forwarding contacts.

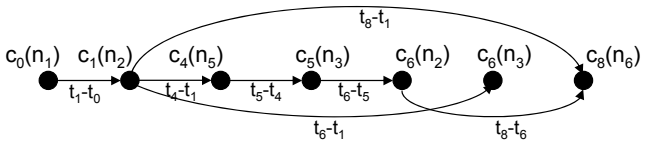

Figure 2: Contact-vertex graph representation of the forwarding contacts' sequence. The edge weights can be used for the computation of the foremost forwarding path to destination node n_6; when replaced with unity, they yield the minimum hopcount path.

We now describe and analyze a graph representation that explicitly accounts for the relative timing of the nodes' contacts. The representation lends to the computation of minimum delay and hopcount forwarding paths for a given message $msg(s, d, t)$ through the reuse of standard shortest path algorithms, *e.g.*, Dijkstra or Depth First Search (DFS)[2]. Input to the graph derivation process are sequences of contact entries with the general format shown in Fig. 1; each entry includes the two nodes that meet and the time of their encounter. The graph derivation proceeds in two steps:

From the original trace to the forwarding contact sequence. The original contacts' sequence is filtered so that only those contacts that can result in message forwarding, hereafter called *forwarding contacts*, are retained. If t_0 is the generation time of a message at source node s for destination node d, then the filtering step excludes: a) all contact entries up to the first entry c_0 after time t_0 involving node s; b) contact entries in the residual trace that do not present a message forwarding opportunity. A contact $c_j = (n_k, n_l, t_j)$ presents a message forwarding opportunity as long as one of the two nodes, n_k and n_l, had earlier the chance to acquire the message through one or more forwarding contacts. In Fig. 1, for example, contacts (n_1, n_2, t_1) and (n_2, n_5, t_4) are forwarding contacts, whereas contacts (n_3, n_4, t_2) and (n_4, n_6, t_7) are not; and c) all contact entries after the first forwarding contact c_d involving node d.

From an implementation point of view, finding the first

[2]Our construct is effectively a simpler alternative to the modified Dijkstra algorithm in [10] and the proposed algorithms in [3],[13] that require explicit formulations of the time-varying edge weight functions in the underlying graph.

contact record $(s, *, t) : t \geq t_0$, requires a binary search with argument t_0 within the full contact entry set. If N is the total number of time-ordered contact entries, the time complexity of this search is $O(logN)$. We then need to read all contact records *after* $(s, *, t)$, which takes $O(N)$ time. To filter out non-forwarding contacts, the process needs to maintain a list of nodes that appear in forwarding contacts (*forwarders' list*). Initially, the list includes only node s and after reading contact c_0 it grows to include the node encountered by s. The list is subsequently updated upon parsing contact entries that involve one of the nodes already listed therein. From complexity point of view, the forwarders' list can be implemented with a binary heap using the node index as key. It involves $O(|V|)$ insert operations, where V is the set of network nodes, costing $O(log|V|)$ time and $O(2N)$ search operations at a cost of $O(log|V|)$ each. Therefore, the overall complexity of the trace-filtering step is $O(Nlog(|V|) + N + logN + |V|log|V|) = O(Nlog|V|)$, which, for typical cardinality values of the node set V, is in the order of the time needed to read the contact records.

Building the forwarding contact graph. The remaining N_r contact entries are then transformed to a directed forwarding contact-vertex graph $G_c = (V_c, E_c)$, where *the set of vertices V_c corresponds to encounters, i.e., node pairs rather than individual nodes.* Each vertex is annotated with the node that acquires the message (one message copy). The sequence of forwarding contacts is again read sequentially and a list of size $O(|V|)$ stores the nodes contributing to the message forwarding (*reduced forwarders' list*). When a contact involving nodes $(n_k,\ n_l)$ is retrieved: a) one new vertex is added if *exactly one* of the encountered nodes already appears in the reduced forwarders' list; b) two new vertices are added if both encountered nodes already appear in the reduced forwarders' list; c) no vertex is added if a contact involving the two nodes has happened before. Moreover, directed edges are drawn towards the new vertex(ices) from *all* existing contact-vertices annotated with the encountered nodes. Hence, in G_c directed edges always point to more recent contacts. The cardinality of the respective contact-vertex set is $|V_c| = O(|V|^2)$ and the edge set $|E_c| = O(|V|^3)$. Since G_c is Directed and Acyclic (DAG), Dijkstra yields the foremost and shortest paths to destination d in $O(|V_c|+|E_c|)$ time, if graph edges are appropriately weighted (see Fig. 2).

5. EXPERIMENTATION RESULTS

Unless otherwise stated, all results shown in this section are obtained for 5.000 randomly generated message instances $msg(s, d, t)$. The default settings for the contact aggregation time window and contact threshold are $T = 6\ hrs$ and $thr_c = 1$, respectively; the resulting edge densities of the static contact graphs are reported in Figure 7.

5.1 Destination-unaware routing: BC *vs.* CBC

Figures 3 and 4 compare the performance of optimal, BC- and CBC-based routing schemes over unweighted contact graphs. As expected, the centrality-based approaches perform worse than the optimal method both in terms of message delay and hops. Up to 30% of messages are trapped and do not reach their destination; and when they do, it costs up to five times more hops and even more than one day delay. The performance lag of centrality-based schemes varies from trace-to-trace and heavily depends on the extent that the mobility patterns of users mix with each other. In

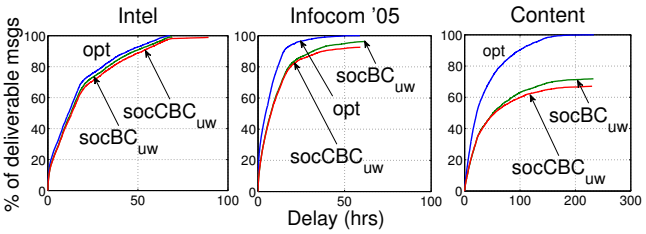

Figure 3: Message delivery delay distribution for destination-aware ($socCBC_{uw}$) and -unaware ($socBC_{uw}$) centrality-based forwarding, T=6hrs

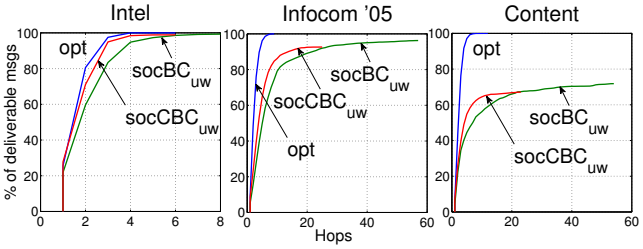

Figure 4: Message hopcount distribution for destination-aware ($socCBC_{uw}$) and -unaware ($socBC_{uw}$) centrality-based forwarding, T=6hrs

the Intel dataset (Fig. 3), involving a small number of processor corporation employees, socioaware routing schemes find close-to-optimal paths. Nodes move in a physically restricted space and meet frequently with eachother (Table 1), as also reflected in the contact graph edge density (Fig. 7). On the contrary, socioaware schemes perform clearly worse over the much sparser Content dataset (Fig 3), involving 36 humans moving around a city-wide area for almost a month.

Less intuitively, the CBC-based forwarding does not outperform the BC-based forwarding with respect to message delivery probabilities and delays. Although the use of CBC implies message routing through more appropriate relays towards the destination, there are numerous graph instances where its interpretation turns out to be problematic. When the 6-hour long aggregation of contacts yields non-connected clusters of nodes, the CBC values of nodes outside the destination's cluster are by default zero. Messages stay long with the source node or are trapped quickly at some intermediate node. On the contrary, with use of the BC metric, nodes take on easier non-zero values and the resulting variance of the metric across nodes lets the message hop from one node to the other. On the other hand, this extra message agility comes at higher cost; under BC-based forwarding, messages end up traversing up to 50% longer paths than under the use of CBC (Fig 4), *i.e.*, messages end up travelling far more in the network. Overall, the use of the CBC metric as routing criterion in forwarding can result in significant energy savings, without degrading severely the delay performance.

5.2 Contact graph representation

Weighted vs. unweighted contact graph: Figure 5 plots the percentage of deliverable messages when the BC metrics are computed over unweighted and weighted graphs and compares them with the reference schemes, *opt* and $PF_{0.2}$.

The results from the five traces suggest that the ranking of the $socBC_{uw}$ and $socBC_w$ schemes is not consistent across all traces. The refined information captured in the weights of the contact graph edges does not always result in more

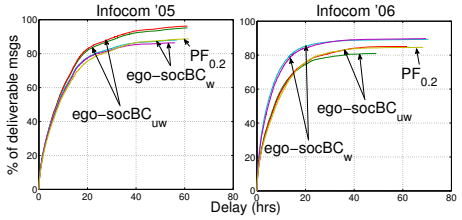

Figure 5: Message delivery delay distrib., T=6hrs

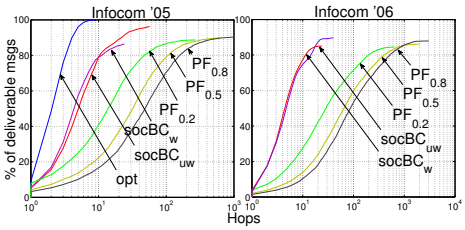

Figure 6: Message hopcount distribution, T=6hrs

effective forwarding decisions. For example, computing BC on the weighted graph representation results in 10% more delivered messages in the first 15 hours of the Infocom'06 experiment (Fig 5), but falls behind $socBC_{uw}$-based routing in the Infocom'05 (Fig 5). This differentiation is reflected in the cumulative delivery probabilities reported in Table 2.

In any case, the differences between schemes that compute centrality over weighted and unweighted graphs do not change the performance trends *across* traces, as discussed in section 5.1. The socioaware schemes again closely approximate optimal performance in the Intel trace, whereas they deviate from it considerably in the Content trace.

Interestingly, the performance of the two BC-aware routing schemes for the two Infocom traces in terms of message delay and percentage of deliverable messages resembles that of the probabilistic forwarding scheme $PF_{0.2}$. The fact that such a socio-agnostic scheme yields comparable results with the 'more-informed' socioaware schemes questions the efficiency of centrality-based decisions. On the other hand, the two BC-aware schemes achieve the same performance at significantly fewer hops compared to those of the probabilistic methods (which have a clear additional energy cost), as shown in Fig. 6. This tradeoff is amplified in the other traces, where the probabilistic scheme delivers up to 20% fewer messages. When the probabilistic forwarding becomes more aggressive, its performance can either approximate closer or even outperform some socioaware schemes, but only at the expense of much more forwarding hops. For p=0.5 and p=0.8, messages traverse up to five times longer paths compared to those of the socioaware methods (Fig. 6).

Impact of contact aggregation window: We now study the sensitivity of centrality-based forwarding approaches to the length of the contact aggregation window, T. T is set to 6 hrs (default value in BubbleRap); we also consider the extreme case of cumulative time window (default SimBetTS implementation), where an edge between two nodes is added to the graph if there has been at least one contact between them at any time in the past.

Intuitively, the routing performance in terms of delay and probability of delivery deteriorates with larger contact aggregation windows. As we aggregate contacts over longer time intervals, the contact graph gets denser and the resolution of the BC metric tends to disappear. In the extreme case of cumulative contact aggregation window, the unweighted contact graph becomes a clique and the centrality-based routing idea is canceled. Furthermore, contacts that occurred rarely in the distant past are aggregated in the graph with the same weight-importance as the recent or regular ones adding noise in the BC computation. On the other hand, the resilience of centrality-based routing is higher when centrality metrics are computed over weighted graphs, where the edge weights capture better the time dimension. This trend is exemplified in the plots of the Cambridge trace (Fig. 8). The ranking of $socBC_w$ and $socBC_{uw}$ curves is reversed as the aggregation window becomes cumulative, with both approaches performing worse in absolute terms than when T=6hrs. Similar results appear across all traces.

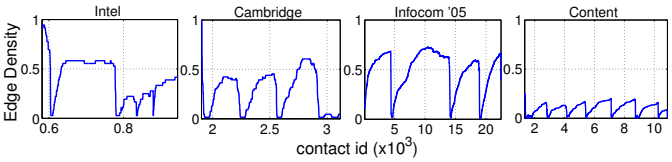

Figure 7: Edge density against contact id

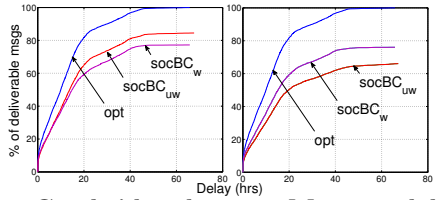

Figure 8: Cambridge dataset: Message delivery delay distribution for BC-based policies, T=6hrs (left) and growing time window (right)

5.3 Socio- *vs.* ego-centric BC computation

The egocentric computation of BC indices is in many cases mandatory in realistic centrality-based routing implementations. Interestingly, our results suggest that the added insights via network-wide information does not significantly benefit the forwarding decisions. It is consistent across all traces (*e.g.*, Fig. 5) that the two metric variants, $egoBC_w$ and $socBC_w$, yield almost identical message delays when computed over weighted contact graphs; whereas, there is a slight advantage of sociocentric BC, $socBC_{uw}$, when centrality is computed over unweighted contact graphs. Similar results hold for the overall probability of delivery shown in Table 2. Using only information about directly encountered nodes to determine centrality penalizes the probability of message delivery only marginally (less than 6% fewer delivered messages in all cases). The highly similar performance of the two BC variants is justified by their strong positive correlation, both in absolute values (Pearson) and ranks (Spearman), across the trace (Fig. 9 and 10). Having similar views about how nodes rank with respect to the BC metric, they end up making similar next-hop forwarding decisions.

Table 2: Probability of message delivery

DataSet	Probability of delivery (6h window)			
	$egoBC_{uw}$	$socBC_{uw}$	$egoBC_w$	$socBC_w$
Intel	99.26	99.26	99.24	99.24
Cambridge	82.68	84.50	82.14	77.24
Infocom'05	95.21	96.25	88.06	86.40
Content	65.55	71.84	69.18	69.20
Infocom'06	80.93	85.08	89.33	89.73

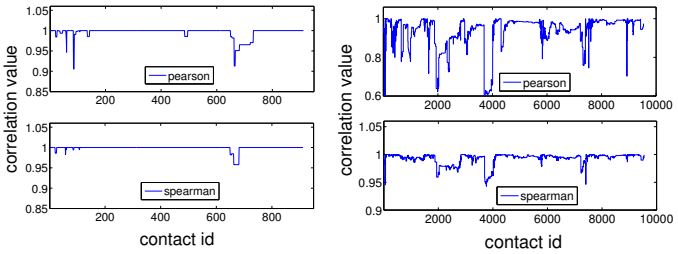

Figure 9: SocioBC-EgoBC correlation over unweighted contact graphs for the Intel(left) and Content(right) datasets

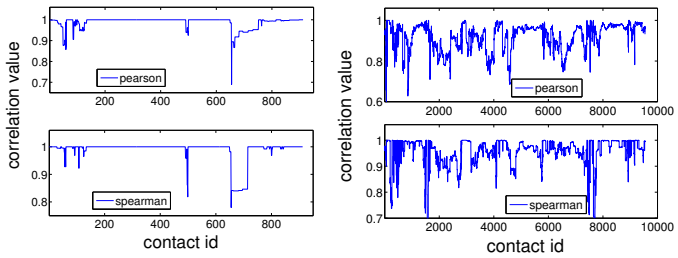

Figure 10: SocioBC-EgoBC correlation over weighted contact graphs for the Intel(left) and Content(right) datasets

6. CONCLUSIONS

We have carried out an experimental study of centrality-based opportunistic routing. Rather than considering a particular protocol implementation, we have assessed the basic routing primitive of centrality-based schemes. We have studied three fundamental alternatives to the sociocentric computation of the original betweenness centrality metric over unweighted contact graphs. The results of our evaluation are summarized into the following points:

- Inline with intuition, the 'greedy' forwarding decisions of centrality-based routing may deviate significantly from the optimal ones and result in considerably worse performance. The latter depends on the way nodes' mobility patterns mix with each other.
- Replacing BC with its destination-aware counterpart does not give benefits in terms of message delay and delivery probability, but can save battery power by reducing the number of message hops in the network.
- Computing BC over weighted graphs does not consistently improve performance for all traces. Nevertheless, the routing protocol is more resilient to variations of the time window used for contact aggregation. In particular, if a cumulative time window is used for contact aggregation, the use of weighted graphs is recommended.
- Interestingly, even the simple scheme of probabilistically forwarding the message upon each encounter yields comparable performance with centrality-based routing albeit at the expense of more message hops in the network.
- Using the egocentric BC variant penalizes routing performance only marginally, when computed over unweighted and even less when computed over weighted graphs. This is further explained by the strong positive correlation between socio- and egocentric BC in almost all traces.

Finally, we have introduced a graph representation that transforms the contact trace to a contact-vertex graph and allows computing foremost and shortest message paths via direct application of well-established shortest-path algorithms. Currently, we are looking closer into the impact of weighted contact graphs on centrality-based routing and expanding our study to "many-to-many" communication settings.

7. ACKNOWLEDGMENTS

This work has been supported in part by the EC-FP7-FET project RECOGNITION (FP7-IST- 257756), the Marie Curie grant RETUNE (FP7-PEOPLE- 2009-IEF-255409), and the National and Kapodistrian University of Athens.

8. REFERENCES

[1] C. Boldrini et al. Hibop: a history based routing protocol for opportunistic networks. In *IEEE WoWMoM*, pages 1–12, 2007.

[2] B. Bui-Xuan et al. Computing shortest, fastest, and foremost journeys in dynamic networks. *Int'l Jnl of Foundations of Computer Science*, 14(2), April 2003.

[3] A. Chaintreau et al. The diameter of opportunistic mobile networks. In *ACM CoNEXT '07*, N.Y., USA.

[4] A. Chaintreau et al. Impact of human mobility on the design of opportunistic forwarding algorithms. In *Proc. IEEE INFOCOM '06*, pages 1–13, April 2006.

[5] E. M. Daly and M. Haahr. Social network analysis for information flow in disconnected delay-tolerant manets. *IEEE Trans. Mob. Comput.*, 8(5), May 2009.

[6] M. Everett and S. P. Borgatti. Ego network betweenness. *Social Networks*, 27(1):31–38, 2005.

[7] M. Grossglauser and D. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Trans. Netw.*, 10(4):477–486, Aug. 2002.

[8] T. Hossmann et al. Know thy neighbor: Towards optimal mapping of contacts to social graphs for dtn routing. In *IEEE Infocom'10*, San Diego, USA, 2010.

[9] P. Hui et al. Bubble rap: Social-based forwarding in delay tolerant networks. *IEEE Trans. Mob. Comput.*, (To Appear) 2011.

[10] S. Jain et al. Routing in a delay tolerant network. *SIGCOMM Comp. Com. Rev.*, 34:145–158, 2004.

[11] A. Lindgren et al. Probabilistic routing in intermittently connected networks. *Lecture Notes in Computer Science*, 3126:239–254, Jan. 2004.

[12] P. Marsden. Egocentric and sociocentric measures of network centrality. *Social Networks*, 24(4):407–422, October 2002.

[13] A. Orda and R. Rom. Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. *Journal of ACM*, 37(3):607–625, July 1990.

[14] P. Pantazopoulos et al. Efficient social-aware content placement for opportunistic networks. In *IFIP/IEEE WONS*, Kranjska Gora, Slovenia, February, 3-5 2010.

[15] T. Spyropoulos et al. Efficient routing in intermittently connected mobile networks: The Multiple-Copy case. *IEEE/ACM Trans. Netw.*, 16(1):77–90, Feb. 2008.

[16] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, Duke University, April 2000.