ONSIDE: Socially-Aware and Interest-Based Dissemination in Opportunistic Networks

Radu-Ioan Ciobanu*, Radu-Corneliu Marin*, Ciprian Dobre*,
Valentin Cristea*, Constandinos X. Mavromoustakis†

*Computer Science Department, Faculty of Automatic Control and Computers
University Politehnica of Bucharest
Bucharest, Romania
{radu.ciobanu, radu.marin}@cti.pub.ro, {ciprian.dobre, valentin.cristea}@cs.pub.ro

†Department of Computer Science
University of Nicosia
Nicosia, Cyprus
mavromoustakis.c@unic.ac.cy

***Abstract*—Data dissemination in opportunistic networks poses a series of challenges, since there is no central entity aware of all the nodes' subscriptions. Each individual node is only aware of its own interests and those of a node that it is contact with, if any. Thus, dissemination is generally performed using epidemic algorithms that flood the network, but they have the disadvantage that the network overhead and congestion are very high. In this paper, we propose ONSIDE, an algorithm that leverages a node's online social connections (i.e. friends on social networks such as Facebook or Google+), its interests and the history of contacts, in order to decrease congestion and required bandwidth, while not affecting the overall network's hit rate and the delivery latency. We present the results of testing our algorithm using an opportunistic network emulator and three mobility traces taken in different environments.**

I. Introduction

Opportunistic networks (or OppNets) have been a main focus of research in the area of mobile networks in the past years. They are composed almost entirely of mobile devices (such as smartphones or laptops) that communicate between each other using a store-carry-and-forward (SCF) paradigm. This implies that nodes are only able to communicate when they are within wireless range of each other. Whenever a destination is not directly accessible, a source opportunistically forwards data to its neighbors. Disconnections between nodes are the norm in opportunistic networks, so forwarding and dissemination algorithms should make the most of a contact, especially if the network is sparse and nodes don't meet often.

Data dissemination in OppNets is a difficult problem: users might dynamically appear and disappear from the network, publishers and subscribers might be completely unaware of each other, and they might never even connect at the same time to the same part of the network. Moreover, there is no central entity that is aware of all the nodes' subscriptions. Instead, each node is only aware of its own interests and those of a node that it is contact with. Therefore, data should be moved and replicated in the network in order to be carried to interested users in spite of disconnections and partitions. A trivial solution would be to epidemically flood the entire network with all the generated data objects, but this would clearly saturate network and device resources. However, context information can be leveraged, in order to decrease the congestion and network overhead. Information about a node's friends from an online social network or about a node's interests can be very helpful in correctly guiding data items towards subscribed nodes. This happens because OppNets are socially-driven. The nodes are humans that adhere to social interaction patterns: a person has a relatively predictable behavior, encountering socially-connected nodes with a high probability. People also tend to group together based on interests, from colleagues working in the same area, to friends getting together at a pub.

978-1-4799-0913-1/14/$31.00

Thus, in this paper we propose ONSIDE (OpportuNistic Socially-aware and Interest-based DissEmination), an algorithm that leverages information about a node's social connections, interests and contact history, in order to decrease network overhead and congestion, while not affecting the network's hit rate and delivery latency. This is done by carefully selecting the nodes that act as forwarders, instead of simply flooding every node. We compare our solution to existing algorithms using three mobility traces.

There are multiple uses for an ONSIDE-based framework in real-life. For example, let's assume that a user A is interested in rock music and a user B is interested in football. The two users are friends and thus meet each other at a pub for a drink. While there, they come in contact with other users carrying mobile devices, each having various interests and carrying mostly data tagged with the corresponding labels. User A would download data items tagged with rock music, while user B downloads information regarding football. After the two users leave the pub, A goes to a rock concert and B goes to a football match. Since A has downloaded rock-related data items and B football-related ones, each of the two users can now spread that information to the other participants at the event they are attending. A will share rock-related information with the other members of the concert audience,

since them attending the concert probably means they are interested in rock music. Similarly, B is able to share the information he has downloaded with the other fans in the stadium. Moreover, the two nodes don't only spread data, but also download information they are interested in from the other users they are in contact with. Using epidemic-based dissemination algorithms for such a situation, where many contacts happen between the event participants, would lead to congestion, since nodes would not only download data they are interested in, but also irrelevant data which would fill their buffers quickly.

II. Related Work

OppNets have been an important research topic in the area of mobile networking in recent years, mainly due to the ubiquitousness of mobile devices of all shapes and sizes. However, the focus has mainly been on routing and forwarding, where a message is sent from a single source node to a unique destination, as opposed to data dissemination, where nodes subscribe to channels that publish data. A thorough analysis of opportunistic networking was done by Conti et al. [1], who analyze functions such as message forwarding, security, data dissemination and mobility models. Several opportunistic forwarding algorithms are reviewed in their article, as well as ContentPlace [2], which is a community-aware algorithm that exploits information about users' social relationships to decide where to place user data.

Since data dissemination presumes sending a message to multiple nodes, Epidemic [3] has been employed in such situations. The algorithm simply floods the network with a message until it reaches all interested subscribers. When two Epidemic nodes meet, they exchange all the messages they carry between each other. This way, assuming that a node can store an unlimited number of messages in its data memory, the maximum hit rate of the network is guaranteed. However, flooding the network with messages can very easily lead to congestion and high overhead, especially in dense networks with high publishing activity.

Data routing and dissemination algorithms have evolved along the years, from simple flooding solutions such as Epidemic, to more complex algorithms that use social metrics, contact prediction or history analysis. Such a routing algorithm is ML-SOR [4], which extracts social network information from multiple contexts, and analyzes encountered nodes in terms of node centrality, tie strength and link prediction on different social network layers. When an ML-SOR node A encounters a node B, it computes a social metric called MLS for all the messages in B's memory, both from its own standpoint, as well as from B's. If MLS is higher for node A, then it sends a download request for that particular message. The social metric is computed based on three components: CS, TSS and LPS. CS represents the centrality of the nodes in the DSN (detected social network layer), while TSS and LPS are computed with regard to the message's destination. TSS is the online social network strength between the analyzed node and the destination, and LPS is a link predictor computed on an interest network layer. Interest-based routing algorithms also include a solution proposed by Moghadam and Schulzrinne [5], where content is spread only to nodes that are interested in it, or SANE [6], where a node also downloads a data item if it is similar to content the node is interested in. The disadvantage of these two solutions is that they are very restrictive in terms of message exchanges, since data is forwarded only to interested nodes. Thus, if a node never encounters another node that is interested in a data item it stores, then that item might not reach any interested node. This is why, as we show in Sect. III, ONSIDE leverages other nodes that are not necessarily interested in a data item, but have a high chance of encountering other nodes that are, to deliver a data item. SocialCast [7] also uses interest information when performing dissemination, combined with prediction information using Kalman filters.

However, these solutions are used for routing, whereas ONSIDE's goal is to offer a publish/subscribe solution. Moreover, ONSIDE also has the advantage of using context information of various types (such as social knowledge, contact history, nodes' interests) with the purpose of reducing congestion and achieving high hit rates.

III. ONSIDE

This section presents ONSIDE, which is an algorithm for socially-aware and interest-based opportunistic data dissemination that attempts to decrease an OppNet's overall bandwidth consumption and reduce its congestion, while not affecting the average channel hit rate and the delivery latency. It is based on several assumptions. Firstly, ONSIDE takes advantage of the fact that nodes that have common interests (i.e. that are subscribed to the same channels) tend to meet each other more often than nodes that do not. This happens because humans generally form groups (i.e. communities) based on similar tastes and preferences, since people sharing common interests are more likely to bond together. This assumption has been shown to be true in various previous works [5], [6], [7]. Because of this, we believe that data dissemination in opportunistic networks can be improved in terms of bandwidth usage and congestion by leveraging interest information when performing dissemination decisions.

The second assumption that ONSIDE is based on states that connections from online social networks (such as Facebook, Google+ or LinkedIn) are respected in an OppNet node's encounters. We have shown in [8] that a node encounters other socially-connected nodes with a high probability. Not only is this true, but there is also a high chance that a node encounters a second-degree neighbor.

Whenever two nodes running ONSIDE meet, they exchange lists of messages in their data memory (characterized by unique IDs and channel information) and lists of topics each node is interested in. Based on this information, each node analyzes the other node's messages and decides which of them should be downloaded. It then sends a download request for those messages, and starts downloading them one by one until it finishes, or until the two nodes are no longer in contact. The function used by a node A to analyze a message M from a node B and to decide whether it should be downloaded is:

$$\begin{aligned} exchange(A,B,M) = (&common\_interests(A,B) \geq 1) \\ &\wedge (interested(A, M.topic) \\ &\vee (interested\_friends(A, M.topic) \geq thr_f) \\ &\vee (interests\_encountered(A, M.topic) \geq thr_i)) \end{aligned} \tag{1}$$

The result of the $exchange$ function is a boolean value that specifies whether a download request should be made to B for message M. The $common\_interests$ function returns the number of topics that both A and B are interested in. This way, data transfers are only performed between nodes with at least one common interest, based on the previous assumption that these nodes encounter each other often and are thus able to successfully deliver channel data to all subscribed nodes. The second component of the $exchange$ function is $interested$, which returns $true$ if node A is subscribed to the channel that generated message M (i.e. if it is interested in M's topic). By using this function, a node will not only download a message for itself and then drop it after use, but will also store it for others, since it is highly likely to encounter other nodes that have similar interests to its own. The $interested\_friends$ function returns the number of online social network friends of node A that are subscribed to the channel that generated M. This component of the $exchange$ function has the role of further reducing the amount of messages exchanged in the network, by only requesting a message if a node's social network friends are also interested in it. This not only reduces the congestion, but also has the role of speeding up the message's delivery. thr_f is a threshold that can be varied according to the density of the OppNet and of the social network. Finally, the last component of the $exchange$ function, $interests\_encountered$, is computed based on node A's history of encounters. It returns the percentage of encounters with nodes that are interested in messages similar to M. This function is based on the assumption that a node's behavior in an OppNet is predictable (as shown in [9]), so that if it encountered many nodes subscribed to a certain channel, it is likely to encounter others in the future as well. thr_i is a threshold between 0 and 1 that can be varied depending on the number of channels in the OppNet.

We believe that ONSIDE is able to outperform other solutions because it allows nodes to download not only data items they are interested in. Thus, nodes can act as relays for other nodes that they are socially connected with, they have common interests with, or they have met recently (and will thus probably meet again soon). For this reason, situations such as islands of social interests can be bypassed, since data items are able to exit these islands by being relayed to uninterested nodes, which act altruistically in order to help their "friends". Therefore, the hit rate is higher than for methods where nodes selfishly download only items of interest to themselves.

IV. Algorithm Analysis

A. Mobility Traces

All of our experiments were performed on three mobility traces that, aside from node contacts, offer information regarding the interests of the participating nodes. The results were obtained by using MobEmu [8], an opportunistic network emulator that is able to replay a trace and apply a desired algorithm when two nodes meet. The three traces used for this analysis are Infocom [10], Sigcomm [11] and UPB [12]. Details about each of them can be found in Table I. Unlike the other two traces we tested with, Infocom has the disadvantage that it doesn't offer information about social connections between the trace participants. Therefore, when testing with this trace, the online social network component of the applied algorithm is not used.

TABLE I. Information about the mobility traces used.

Trace	Nodes	Duration	Type	Topics	Topics / node
Infocom	98	4 days	Conference	27	14.53
Sigcomm	76	4 days	Conference	154	15.61
UPB	66	64 days	Academic	5	3.51

B. Experimental Setup

In our experiments, data (in the shape of messages) is generated through channels that nodes are able to subscribe to. When a node is subscribed to a channel, it is interested in any data generated by that channel that it hasn't received yet. We consider that a channel is represented by a topic of interest, so there are 27 channels for Infocom, 154 for Sigcomm and 5 for UPB (i.e. the total number of topics for each trace). Every node interested in a certain topic can generate information on the corresponding channel, but not on channels that don't match its interests. Each node that has at least one interest generates 30 messages per day in the two-hour interval when the most contacts happen (which, for all three traces, is close to midday). A node that is interested in multiple topics is able to generate data for each of the corresponding channels, by choosing randomly out of its interests.

In order to highlight the benefits of ONSIDE, we compare it to existing data dissemination solutions for OppNets. Thus, we have also implemented Epidemic in MobEmu, which allows us to see the maximum available hit rate for each trace, to verify how close ONSIDE gets to achieving it. However, since the classic Epidemic algorithm is not entirely feasible in real life (given that it assumes an unlimited data memory), we also compare ONSIDE to a limited-memory version of Epidemic (Limited Epidemic), that behaves exactly like the original implementation, except that, when the data memory is full and a new message should be downloaded, an existing message must be deleted from memory.

Aside from Epidemic, we also compare ONSIDE to a dissemination-modified version of ML-SOR. We have chosen this algorithm since it is also interest-based and socially-aware like ONSIDE. The basic ML-SOR algorithm, as described in Sect. II, computes a social metric MLS as a function of three utility scores: CS, TSS and LPS. Both TSS, as well as LPS, are computed based on information about the two encountering nodes and the destination of the current message. However, in data dissemination we can't talk about a message's destination, since there isn't a single destination, and the node that generates the message isn't aware of the channel's subscribers. Therefore, we have modified ML-SOR to compute the social metric using information about the source of the message, instead of the destination. This means that, regarding the TSS component, when a node A encounters a node B, it will forward a message M to it if B is socially connected to M's source on more online social networks than A. This is based on the assumption that a node is more likely to encounter nodes from its social community than regular nodes, and that nodes from the same community share common interests. Similarly, when analyzing the LPS component, a node A will

forward a message M to B if B has more interests in common with M's source than A does. This is plausible, since a node is likely to encounter nodes sharing its interests (as shown in Sect. III), so B has a better chance of further disseminating M than A does. The third ML-SOR component (CS) remains unmodified, since it is only computed in regard to A and B.

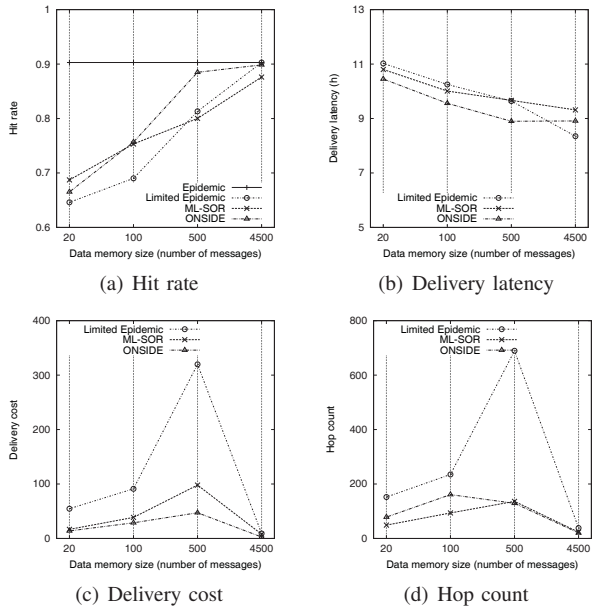
In order to analyze how the various algorithms we test with behave in different conditions, we vary a node's data memory size. Thus, a node is able to store either 20, 100, 500 or 4500 messages at once in its memory. Assuming that a message has an average size of 5 MB, this means that the range of node memory we test with is 100 MB - 22 GB. We consider these to be appropriate values for different kinds of mobile devices, ranging from older smartphones to top-of-the-line devices. For all three memory-limited algorithms we test with (i.e. Limited Epidemic, ML-SOR and ONSIDE), whenever a node decides to download a message but its data memory is full, it deletes the oldest stored message and replaces it with the new one.

As shown in Sect. III, ONSIDE has two thresholds: thr_f for the number of friends interested in a topic, and thr_i for the percentage of encounters with nodes interested in a certain topic. These values are different for each trace. For Sigcomm, thr_f is 1 and thr_i is 0.75, while for UPB, thr_f is 5 and thr_i is 0.2. The reason thr_f is higher for UPB is that, since the trace is taken in an academic environment instead of a conference, most of the participants are socially-connected since they are colleagues. Having a low thr_f would lead to more messages being exchanged, and thus the potential of congestion. Finally, since Infocom doesn't contain information about the social network, thr_f is 0 and thr_i is 0.3. thr_i depends on the average number of topics per node and the total number of topics available, as shown in Table I. When the topics per node - total topics ratio is high, the threshold should be lower, since otherwise there would be too many message exchanges.

When analyzing the results, we focus on hit rate, delivery latency, hop count and delivery cost. The hit rate is the ratio between the number of messages that have successfully arrived at nodes subscribed to the corresponding channels, and the total number of messages generated, each of them multiplied by the number of subscribers to the channel. The delivery latency is the average amount of time passed between the generation of a message and its delivery to a subscribed node. The hop count is the number of nodes that carried a message until it reached its destination on the shortest path, and the delivery cost is the ratio between the total number of messages exchanged and the number of generated messages multiplied by the number of corresponding channel subscribers. These last two metrics are measures of network and node congestion.

C. Results

Figure 1 shows the results obtained by applying Epidemic, Limited Epidemic, ML-SOR and ONSIDE on the Sigcomm trace. It can be seen in Fig. 1(a) that ONSIDE generally performs better than both ML-SOR, as well as Limited Epidemic (for a 500-message data memory, the ONSIDE hit rate is 7% better than Limited Epidemic's and 8% better than ML-SOR's). For a data memory that can store 4500 messages, ONSIDE yields a hit rate that is very close to the maximum value (which is obtained when running Epidemic and Limited

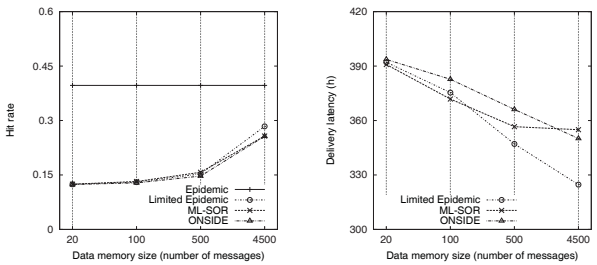


(a) Hit rate (b) Delivery latency

(c) Delivery cost (d) Hop count

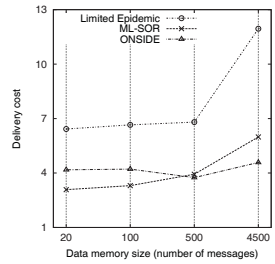
Fig. 1. Results for the Sigcomm trace.

Epidemic). Limited Epidemic is able to achieve maximum hit rate because the memory size is large enough to fit all the messages generated in the trace, since Sigcomm's duration is only three days. The delivery latency charts, presented in Fig. 1(b), show similar results: ONSIDE is able to achieve a better delivery latency than ML-SOR regardless of the data memory size (with a maximum improvement of up to 1.7 hours), whereas Limited Epidemic only outperforms our algorithm when the data memory is large enough to store all the messages generated in the trace. However, the downside of Epidemic-based algorithms is evident from Fig. 1(c) and Fig. 1(d), where it can be seen that the bandwidth used and the network and node congestion are really high. ONSIDE's delivery cost is lower than the ones obtained by Limited Epidemic and ML-SOR for all data memory sizes, yielding an improvement of up to 272 less messages exchanged per generated message compared with Limited Epidemic and 40 when compared to ML-SOR (for a data memory that can store 500 messages). Since there are 25,998 total data items generated in the network (value obtained by counting the number of interested nodes for each message), ONSIDE manages to decrease the total number of data items transferred during the trace by 7,086,015 compared to Limited Epidemic and by 1,320,958 compared to ML-SOR. Assuming a data item has 5 MB, this means that using ONSIDE leads to transferring 32 and 12 less TB in three days. Regarding hop count, ONSIDE again clearly outperforms Limited Epidemic for all data memory sizes, but it does achieve a higher hop count than ML-SOR for lower data memory sizes (like 20 and 100). This shows that ML-SOR might be a bit more suitable than ONSIDE for older devices that have more limited memories, but these days a smartphone's Flash storage is in the order of GBs. It is interesting to note that, in both Fig. 1(c) and Fig. 1(d), there is a large spike when a node's data memory can store 500 messages. This happens because the memory is large enough to store messages that are of interest to various types of nodes (meaning that it has to perform many forward

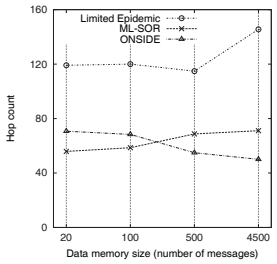


(a) Hit rate

(b) Delivery latency

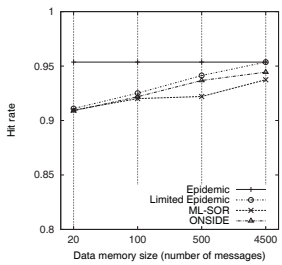


(c) Delivery cost

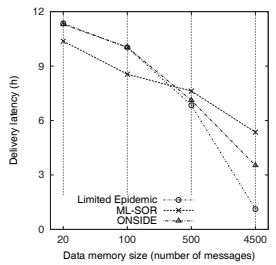


(d) Hop count

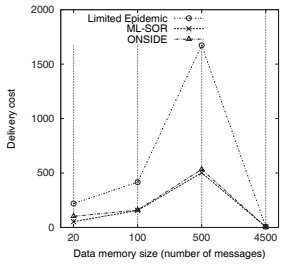
Fig. 2. Results for the UPB trace.



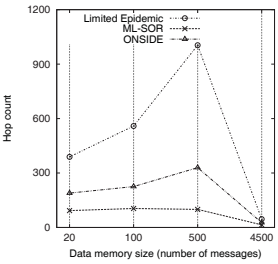
(a) Hit rate



(b) Delivery latency



(c) Delivery cost



(d) Hop count

Fig. 3. Results for the Infocom trace.

operations), but not large enough to contain all the messages generated in the trace.

The UPB results are shown in Fig. 2. Since the duration of the trace is much higher than Sigcomm's, there are a lot more messages generated in the network, so the maximum hit rate is harder to achieve with a limited data memory. This is clear from Fig. 2(a), where even Limited Epidemic doesn't manage to achieve a very high hit rate. The ONSIDE algorithm performs similarly to both Limited Epidemic, as well as ML-SOR. Because this trace has a much longer duration that Sigcomm and the network is relatively sparse (with not many contacts), the delivery latency values are very high. Regardless of this, the values (shown in Fig. 2(b)) are similar for all three algorithms, with a slight edge for Limited Epidemic for higher data memory sizes. However, this comes with the cost of increased congestion and overhead, as seen in Fig. 2(c) and Fig. 2(d), where Limited Epidemic performs much worse than ONSIDE and ML-SOR. Regarding delivery cost, ONSIDE performs the best out of all three algorithms for data memory sizes of 500 and 4500 messages, leading to improvements of up to 7.5 (i.e. 7.5 less messages are exchanged per generated message). For lower memory sizes, ML-SOR performs better, which further proves that it is better suited for older devices with small memories. The situation regarding hop count is identical: ONSIDE has the better results for memories of 500 and 4500 messages, while ML-SOR is better for a lower memory size. Because there are much more messages in the network than at Sigcomm, the spike in delivery cost and hop count that was seen at the previous trace doesn't appear here. Instead, the delivery cost and hop count keep on growing, since a node is never able to store all the messages from the network. The disadvantage of ONSIDE for UPB is that there are only five very broad topics of interest reported by the trace. This means that there is a very high chance that, when two nodes encounter each other, they have at least one interest in common. This leads to a lot of data exchanges in ONSIDE, and thus older messages being deleted from memory.

Finally, the results for the Infocom trace are presented in Fig. 3. As stated before, Infocom has the disadvantage of not containing social information about the participating nodes. Therefore, the TSS component from ML-SOR will always be 0. Similarly, ONSIDE doesn't use the $interested\_friends$ component, since thr_f is set to 0. Figure 3(a) shows that ONSIDE yields better hit rates than ML-SOR for all data memory sizes except 20, keeping close to the two Epidemic versions, especially when the data memory size is higher. Regarding delivery latency, ONSIDE fares better than ML-SOR for large memory sizes. Limited Epidemic has the best behavior in terms of latency, but it pays for it with a higher degree of congestion, as seen in Fig. 3(c) and Fig. 3(d). The hop count obtained by Limited Epidemic is extremely large, about three times larger than the one ONSIDE yields. However, ML-SOR performs better in terms of hop count than both the other two solutions. The delivery cost is lower for ML-SOR and ONSIDE, with our solution faring much better when the data memory size is higher. Similar to the Sigcomm results, the spike in delivery cost and hop count at 500 messages appears again, because this is a short trace and thus, for a data memory of 4500 messages, a node is able to store all the messages in the network.

Therefore, the results show that, generally, both ONSIDE and ML-SOR barely affect the overall hit rate (with a slight advantage for ONSIDE), while decreasing the congestion and overhead. For most of the situations, ML-SOR seems to work better for lower data memory sizes, while ONSIDE fares well for nodes that are able to store more messages. This happens because ONSIDE has a higher point of balance, when the messages stored in a node's memory reach a local maximum in terms of utility, making subsequent data transfers unnecessary. For ML-SOR, this point is lower, but the equilibrium is less stable, finally leading to frequent data exchanges.

We believe, based on the results shown here, that ONSIDE would be able to successfully deliver data while avoiding congestion even in OppNets formed of tens of thousands

of nodes. One way to do that would be to have a much finer-grained approach to representing a user's interests, so the number of interest-based communities would be larger. This way, the average number of nodes per interest would remain low, and thus the number of exchanged message would only increase for users that have many interests and implicitly belong to a great number of communities.

V. Conclusions

In this paper, we have presented ONSIDE, a socially-aware and interest-based data dissemination algorithm for opportunistic networks. Through experimental results, we have shown that for traces taken in conference environments, where there are many nodes spread over a small surface, ONSIDE is able to reduce congestion without affecting the hit rate and delivery latency, when compared to other solutions such as Epidemic or ML-SOR. For older types of OppNet devices, that have lower memory sizes, ML-SOR tends to perform slightly better in terms of congestion. However, nowadays the mobile devices employed in OppNets have at least 1 GB of Flash storage, so ONSIDE would be better employed. We have also tested on a mobility trace with a longer duration (UPB), taken in an academic environment, and have shown that ONSIDE's performance in terms of congestion is only slightly better than ML-SOR's. This is caused by the fact that the UPB trace has a very coarse-grained representation of interests, and thus the interest-based communities tend to be larger, leading to many message exchanges. As future work, we wish to be able to infer interests based on the behavior of the nodes in the network.

The three traces presented here aren't necessarily representative of all types of OppNets, since they are biased because of the fact that all the participants are part of an academic environment, and thus naturally have many interests in common. Moreover, the durations of the traces are not very long, and the physical spaces where the nodes roam are also relatively small. For this reason, we plan on also testing our solution on larger traces, taken in different environments. However, most of the traces available online don't contain context information used by ONSIDE, such as node interests or social connections. Consequently, methods of inferring node interests (such as [13]) or communities (such as k-CLIQUE [14]) will have to be employed for those traces.

We also plan on testing different heuristics for sorting the messages in a node's memory. This way, when the memory is full and a new data item should be downloaded, the node will remove the least important message, thus increasing the hit rate and delivery latency. We also wish to analyze the problem of selfishness in OppNets, and to add capabilities of detecting and penalizing nodes that don't cooperate. Finally, the long-term plan is to extend our research with algorithms that are able to keep data items close to where interested nodes are. The idea is similar to floating content [15], but instead of geographically defining an anchor zone where an object is available, we would define a set of criteria that should be true in order for a certain data item to be kept alive.

Acknowledgment

The research is supported by CyberWater grant of the Romanian National Authority for Scientific Research, CNDI-UEFISCDI, project number 47/2012. This work was also supported by COST Action IC1303: Algorithms, Architectures and Platforms for Enhanced Living Environments (AAPELE).

References

[1] M. Conti, S. Giordano, M. May, and A. Passarella, "From opportunistic networks to opportunistic computing," *Comm. Mag.*, vol. 48, pp. 126–139, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1866991.1867009

[2] C. Boldrini, M. Conti, and A. Passarella, "ContentPlace: social-aware data dissemination in opportunistic networks," in *Proceedings of the 11th international symposium on Modeling, analysis and simulation of wireless and mobile systems*, ser. MSWiM '08. New York, NY, USA: ACM, 2008, pp. 203–210. [Online]. Available: http://doi.acm.org/10.1145/1454503.1454541

[3] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," 2000.

[4] A. Socievole, E. Yoneki, F. De Rango, and J. Crowcroft, "Opportunistic message routing using multi-layer social networks," in *Proceedings of the 2Nd ACM Workshop on High Performance Mobile Opportunistic Systems*, ser. HP-MOSys '13. New York, NY, USA: ACM, 2013, pp. 39–46. [Online]. Available: http://doi.acm.org/10.1145/2507908.2507923

[5] A. Moghadam and H. Schulzrinne, "Interest-aware content distribution protocol for mobile disruption-tolerant networks," in *World of Wireless, Mobile and Multimedia Networks Workshops, 2009. WoWMoM 2009. IEEE International Symposium on a*, June 2009, pp. 1–7.

[6] A. Mei, G. Morabito, P. Santi, and J. Stefa, "Social-aware stateless forwarding in pocket switched networks," in *INFOCOM, 2011 Proceedings IEEE*, April 2011, pp. 251–255.

[7] P. Costa, C. Mascolo, M. Musolesi, and G. Picco, "Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks," *Selected Areas in Communications, IEEE Journal on*, vol. 26, no. 5, pp. 748–760, June 2008.

[8] R. I. Ciobanu, C. Dobre, and V. Cristea, "Social aspects to support opportunistic networks in an academic environment," in *Proceedings of the 11th international conference on Ad-hoc, Mobile, and Wireless Networks*, ser. ADHOC-NOW'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 69–82. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31638-8_6

[9] R. I. Ciobanu and C. Dobre, "Predicting encounters in opportunistic networks," in *Proceedings of the 1st ACM Workshop on High Performance Mobile Opportunistic Systems*, ser. HP-MOSys '12. New York, NY, USA: ACM, 2012, pp. 9–14. [Online]. Available: http://doi.acm.org/10.1145/2386980.2386983

[10] P. Hui and J. Crowcroft, "Bubble Rap: forwarding in small world DTNs in ever decreasing circles," University of Cambridge Computer Laboratory, Tech. Rep. UCAM-CL-TR-684, May 2007. [Online]. Available: http://www.cl.cam.ac.uk/TechReports/UCAM-CL-TR-684.pdf

[11] A.-K. Pietiläinen, E. Oliver, J. LeBrun, G. Varghese, and C. Diot, "MobiClique: middleware for mobile social networking," in *Proceedings of the 2nd ACM workshop on Online social networks*, ser. WOSN '09. New York, NY, USA: ACM, 2009, pp. 49–54. [Online]. Available: http://doi.acm.org/10.1145/1592665.1592678

[12] R.-C. Marin, C. Dobre, and F. Xhafa, "Exploring Predictability in Mobile Interaction," in *Emerging Intelligent Data and Web Technologies (EIDWT), 2012 Third International Conference on*. IEEE, 2012, pp. 133–139. [Online]. Available: http://dx.doi.org/10.1109/EIDWT.2012.29

[13] A. Noulas, M. Musolesi, M. Pontil, and C. Mascolo, "Inferring interests from mobility and social interactions," in *Proceedings of Workshop on Analyzing Networks and Learning with Graphs (NIPS 2009)*, 2009.

[14] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft, "Distributed community detection in delay tolerant networks," in *Proceedings of 2nd ACM/IEEE international workshop on Mobility in the evolving internet architecture*, ser. MobiArch '07. New York, NY, USA: ACM, 2007, pp. 7:1–7:8. [Online]. Available: http://doi.acm.org/10.1145/1366919.1366929

[15] J. Ott, E. Hyytia, P. Lassila, T. Vaegs, and J. Kangasharju, "Floating content: Information sharing in urban areas," in *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*, 2011, pp. 136–146.