2013 IEEE 27th International Conference on Advanced Information Networking and Applications

Reducing Congestion for Routing Algorithms in Opportunistic Networks with Socially-Aware Node Behavior Prediction

Radu Ioan Ciobanu, Ciprian Dobre, Valentin Cristea
University Politehnica of Bucharest
Bucharest, Romania
radu.ciobanu@cti.pub.ro, ciprian.dobre@cs.pub.ro, valentin.cristea@cs.pub.ro

***Abstract*—Since mobile devices nowadays have become ubiquitous, several types of networks formed over such devices have been proposed. One such approach is opportunistic networking, which is based on a store-carry-and-forward paradigm, where nodes store data and carry it until they reach a suitable node for forwarding. The problem in such networks is how to decide which the next hop will be, since nodes do not have a global view of the network. An inefficient opportunistic routing algorithm can lead to the congestion of a network, because same groups of nodes send messages between each other, without the destination actually receiving the data (or receiving it with a high delay). We describe here a routing algorithm for opportunistic networks that avoids congestion and the overcrowding of nodes, by routing messages only to nodes that have a high chance of reaching a message's destination. This is performed with the use of social networks and node behavior prediction. We show that our algorithm outperforms existing algorithms such as BUBBLE Rap in terms of delivery cost and hit rate, as well as the rate of congestion introduced in the networks.**

***Keywords*-opportunistic; social; prediction; congestion;**

I. Introduction

The rise in number of mobile devices has brought the advent of opportunistic networks, which consist mainly of human-carried mobile devices (e.g. smartphones, tablets, etc.) that are unaware of any network infrastructure and interact with each other based on a store-carry-and-forward paradigm. Nodes communicate opportunistically when they are within wireless range of each other. A node stores data in the form of messages, which are carried around until a node with a higher chance of delivering them to the destination is met. Then, the messages are forwarded to the encountered node. Routes between nodes are dynamically created, and nodes can be opportunistically used as a next hop for bringing each message closer to the destination.

The nodes of an opportunistic network are mobile devices usually carried by people, which are organized into communities according to common professions, workplaces, interests, etc. Generally, members of the same community interact with each other more often than with members of outside communities, so a good opportunistic network routing algorithm should take community organization into consideration. We show here that adding knowledge about social links between opportunistic network nodes to routing and dissemination algorithms greatly improves their effect. We strongly believe that social network connections are a better approximation of human relationships than existing community detection algorithms. However, we show that only using social information about the nodes in the network is not enough to obtain satisfactory results, therefore we attempt to predict the future behavior of a node using the history of encounters and information about its social community. We approximate a node's contact history as a Poisson distribution and use the result in creating a routing algorithm entitled Socially-Aware Prediction (SAP).

When routing in an opportunistic network, messages are generally replicated instead of being moved from one node to another. This can lead to a large number of copies of the same message in the network, which results in congestion of the network. Furthermore, nodes may receive data faster than they are able to forward it, leading to their data memory becoming full. In such situations, nodes must drop some of the messages they were carrying, and the question that arises in this case is how to decide which messages to drop. The main contribution of this paper is that, aside from proposing a new opportunistic routing algorithm, it shows that said algorithm can solve the problem of congestion by transferring data between nodes only when there is a high probability that a node will encounter the destination of a message. Moreover, when a node's buffer becomes full, messages are dropped according to utility values computed for each of them, which signify the importance of the message in terms of future encounters with the destination.

The rest of the paper is structured as follows. Section II presents related work in the field of routing and dissemination in opportunistic networks. In Sect. III we describe SAP. Section IV details the problem of congestion in opportunistic networks and presents contributions in this area. Section V describes the experiments performed and shows the results obtained when comparing SAP to BUBBLE Rap in terms of congestion. Finally, Section VI concludes the paper.

II. Related Work

Since opportunistic networks have become more and more popular over the past years, partly due to the ubiquitousness of mobile devices, several authors have treated this research

1550-445X/13 $26.00 © 2013 IEEE
DOI 10.1109/AINA.2013.63

area in great detail. A review of opportunistic networking can be found in [10], where functions such as message forwarding, security, data dissemination and mobility models are analyzed. Several opportunistic forwarding algorithms are also reviewed, among them being BUBBLE Rap [11], PROPICMAN [16] or HIBOp [4]. We propose a taxonomy for data dissemination algorithms in [7], where we split such algorithms into four main categories. The first category refers to the infrastructure of the network, i.e. the way the network is organized into an overlay. The dissemination techniques are also split according to their node properties, such as state or interaction. The third category of the taxonomy is represented by content characteristics, i.e. the way content is organized and analyzed, and finally the last category is social awareness.

Several other authors propose dissemination algorithms for opportunistic networking. For example, the Socio-Aware Overlay algorithm [21] creates an overlay for an opportunistic network with publish/subscribe communication, composed of nodes with high values of centrality. Another dissemination algorithm, Wireless Ad Hoc Podcasting [15], has the purpose of wireless ad hoc delivery of content among mobile nodes and enables the distribution of content using opportunistic contacts whenever podcasting devices are in wireless range. ContentPlace [5] facilitates data dissemination in resource-constrained opportunistic networks by making content available in regions where interested users are present. In order to optimize content availability, it exploits information about users' social relationships to decide where to place user data. Nodes from ContentPlace use a utility function in order for each node to associate a value to any data object. When a node encounters a peer, it computes the utility values of all the data objects stored in the local and in the peer's cache and then it selects the set of data objects that maximizes the local utility of its cache.

The addition of social network information to opportunistic routing has been studied in [1], where the authors show that using Facebook information instead of community detection algorithms decreases the delivery cost and produces comparable delivery ratio. In [14], an analytical model for the expected hop count and latency of messages delivered in a socially-aware opportunistic routing algorithm is proposed, where the forwarding process is modelled as a semi-Markov process. A socially-aware middleware that learns information about the nodes in the network and then uses it to predict their future movement is proposed in [3]. The middleware was integrated with the Haggle architecture and was used for content sharing, yielding up to 200% improvement in terms of hit rate and 99% reduction in resource consumption in terms of traffic in the network.

The problem of predicting the future behavior of nodes in delay-tolerant and opportunistic networks is also treated in several papers. In [13], a framework for evaluating routing algorithms for DTNs is proposed and the performance of several such algorithms is analyzed in terms of the amount of knowledge about the network that they require. In [19], the authors analyze the predictability of human behavior and mobility on user traces obtained from mobile carriers. In [12], the behavior of a time series is modelled as a Poisson process model and then is modulated using a hidden Markov process. The authors show that using a Poisson model is significantly more accurate at detecting future behavior and known events than a traditional threshold-based technique. Since contact information in an opportunistic network is also a time series, we believe that it can also be approximated as a Poisson distribution, which we prove in [8].

III. Socially-Aware Prediction

We propose here an algorithm for routing data in opportunistic networks, entitled Socially-Aware Prediction (SAP). In the following subsections we describe its components and the way they are combined.

A. Social Connections in Opportunistic Networks

The first component of the SAP algorithm is based on the knowledge that most of the nodes in an opportunistic network are devices carried by people. Therefore, we consider that relationships between carriers should be taken into consideration when routing, since it has been proven that a node is more likely to interact with members of its own social community then with any other nodes [6]. We have also shown that this is true in [9]. In order to test our assumptions in a real-life situation, we collected a mobility trace (UPB 2011) at the University Politehnica of Bucharest using an Android application called Social Tracer. For this purpose, a tracing experiment was performed at the university that lasted for 25 days and had 22 participants. We analyzed the trace in terms of contacts between nodes and showed that the most popular nodes in terms of social relationships had more contacts than regular nodes. We also proved that adding social information from Facebook to distributed BUBBLE Rap [11] instead of detecting social communities on-the-fly using k-CLIQUE improved the hit rate by as much as 10%. We ran tests on another trace as well, corresponding to a larger and more open environment, with less regularity, less contact opportunities and smaller contact durations entitled St. Andrews [1], which was collected using a mobile sensor network with Tmote Invent devices carried by 27 members of the University of St. Andrews for 79 days. In this case, the hit rate was improved by 5.21%. However, other metrics such as delivery latency, hop count or delivery cost for both traces had generally higher values when applying the social changes, with slight exceptions. Thus, we considered it necessary to add an extra component to opportunistic routing in addition to social knowledge, in order to improve these values as well.

B. Predicting Future Node Behavior

That extra component was node prediction. We proposed in [8] a way to predict a node's behavior by analyzing a mobility trace in terms of a node's past encounters and approximating the time series obtained as a Poisson distribution. In order to prove that this is indeed possible, we recorded another mobility trace at the University Politehnica of Bucharest, entitled UPB 2012. It had 66 participants from the faculty and it lasted for 64 days, recording contacts on Bluetooth and WiFi. We also tested on the St. Andrews trace previously described because we wished to show that SAP behaved well in a different situation too, where nodes were not grouped together and the area of the trace encapsulated an entire town, not just a small space.

Since the nodes in our trace were students and teachers at the University Politehnica of Bucharest, their behavior would be predictable because the participants had a fixed daily schedule and interacted with each other at fixed times in a day. We used Shannon's entropy to verify predictability and reached the conclusion that having a contact at the next period of time was mostly predictable, because the entropy obtained was always lower than 0.35, while predicting the node that would be seen at the next encounter yielded entropy values that were as high as 4.25. Reaching these conclusions, we decided to approximate the time series of a node's encounters as a Poisson distribution. In order to prove that such a distribution applied to our trace, we used Pearson's chi-squared test [20] and applied it for every node in the network individually using a one-hour time interval. The final results showed that only 2.49% out of all the hypotheses were rejected. We also applied the chi-squared tests using only unique contacts, i.e. the number of contacts in an hour was considered equal to the number of different nodes encountered in that hour. The results showed that just 1.31% of distributions were not Poisson according to the chi-squared tests. In addition, we ran the tests on the UPB 2011 trace as well and the results for unique contacts were even better than for UPB 2012, since only 0.11% of the hypotheses were rejected. However, the traces performed in the campus of our faculty could have been a particular case, meaning that we couldn't generalize our assumptions yet. Therefore, we also tested our theory on the St. Andrews [1] trace, and the results we obtained showed that just 12.39% of the chi-squared hypotheses were rejected.

To further prove our assumptions, we eliminated the last two weeks from the trace and computed the Poisson distribution probabilities for each hour per day of the week on the remaining series. We compared the values that had the highest Poisson probability (i.e. the most likely values according to the distribution) with the real values. If the Poisson predictions were to be correct, then the two values would be equal. The results of this test showed that, for total encounters 97.59% of the Poisson-predicted values were correct for the next to last week, and 98.42% for the last. When taking into account individual encounters, the predictions were even better (98.24% and 99.14%).

C. The Algorithm

Based on the conclusions drawn in [9] and [8], we propose an algorithm (SAP - Socially-Aware Prediction) that combines socially-aware routing with future node behavior prediction. SAP's main goal is to achieve a better hit rate than existing algorithms do, while reducing delivery cost and hop count (and thus bandwidth consumption and network congestion). However, in this paper we are only concerned with the reduction of network and node congestion.

The SAP algorithm considers that a node possesses a data memory and a cache memory. The actual messages are stored in the data memory, while the cache memory is used to store contact history information. When two nodes are within wireless range of each other, they exchange information about the messages they carry. This information includes a hash of the message's content (which acts as a unique ID), the source and destination together with the communities they belong to, the generation time of the messages, and the number of hops it has traversed so far. Based on this information, each node computes utilities for the messages carried by itself, as well as the neighboring node. It then chooses those with the highest utility values (limited by the size of the data memory) by sending a request to the encountered node with hash values of the required messages. The neighbor sends the messages until the two nodes are no longer in range or all the required messages have reached their target.

The main contribution of the SAP algorithm is its utility function, which is based on the prediction of the future encounters of a node by using its contact history and social network information. SAP computes the utility u of a message M at node A as:

$$u(M, A) = w_1 * U_1(M, A) + w_2 * U_2(M, A)$$

In the formula, w_1 and w_2 are weight values which follow the conditions that $w_1 + w_2 = 1$ and $w_1 > w_2$, while U_1 and U_2 are utility components computed according to the following formulas:

$$U_1(M, A) = freshness(M) + p(M, A) * (1 - \frac{enc(M, A)}{24})$$

$$U_2(M, A) = c_e(M, A) * \frac{s_n(M) + hop(M) + pop(A) + time(M, A)}{4}$$

For U_1, the $freshness$ value can either be 0 if message M is old, or 0.5 if the message has been generated recently. We use it in order for newly-generated messages to start travelling through the network faster. This leads to a higher chance of delivering them to their destinations, since they reach more nodes. $p(M, A)$ is the probability of node A being able to deliver message M. Its computation starts with

the analysis of the cache memory, counting how many times node A encountered each of the other nodes. If a node has been previously met in the same day of the week or in the same two-hour interval as the current time, the total encounters value is increased by 1. For the nodes encountered in the past that are in the same social community as node A, the total number of contacts is doubled. The reason 1 is added to the total number of encounters for nodes that have been met in the same day of the week or in the same two-hour interval when computing $p(M, A)$ is done because there is a certain regularity in the activity of nodes in an academic environment. Therefore, there should be a higher probability of encountering nodes that have been seen in the same intervals. There is a similar reason for doubling the total number of contacts when the nodes met in the past are in the same community as node A, since a node tends to interact more with other members of its own social community. Having the number of encounters for each node, we can compute the probabilities of encountering them by performing a ratio between the number of encounters per node and the total number of encounters.

The next step consists of computing the number of encounters N that node A will have for each of the next 24 hours by using the Poisson probabilities and choosing the value with the highest probability as N. We pick only the first N nodes as potential future contacts for each of the next 24 hours (sorted by probability), and for the rest of them $p(M, A)$ is set to 0. The second component of the product uses $enc(M, A)$, which is the hour of the day when the destination of message M will be met according to the probabilities previously computed. If the destination will not be encountered, then $enc(M, A)$ is set to 24. We multiply $p(M, A)$ by $1 - \frac{enc(M,A)}{24}$ because the sooner a good target for a message is met, the sooner the node can delete the message from its memory and have room for others.

The second component of the utility value is U_2. Here, $c_e(M, A)$ is set to 1 if node A is in the same community as the destination of message M or if node A will ever encounter a node that has a social relationship with M, and 0 otherwise. The information computed for U_1 is used to analyze the potential future encounters of a node. The $s_n(M)$ component is 1 if the source and destination of M are not socially connected, because if a message doesn't have the source and destination in the same community, the chance of it being delivered by the source is low since it will mostly meet messages in its community. Therefore, the messages should be given to a different node that has the chance of reaching the destination community. $hop(M)$ is the normalized number of nodes that M has visited, $pop(A)$ is the normalized popularity value of A according to its social network information (i.e. number of Facebook friends in the opportunistic network), and finally $time(M, A)$ is the total time spent by node A in contact with M's destination. The hop and $time$ values are used because nodes should travel as little as possible before reaching their destination.

We have put the condition $w_1 > w_2$ because messages destined for nodes that will be met in the future are more important than the others, since we know (with a certain probability) that they will be delivered eventually. While it may seem complicated to choose appropriate values for the two weights, in our experiments and tests we empirically observed that the best results were obtained when $w_1 = 0.7$ and $w_2 = 0.3$. However, we are also investigating possible methods to predict the most suitable weights according to the behaviour of the network.

IV. Congestion in Opportunistic Networks

Since data dissemination and routing in an opportunistic network are generally performed through replication, an increased number of message copies may congest the network. This makes it harder for other nodes to communicate, leading to lost messages that will never reach their intended destinations. Moreover, when a node receives data faster than it can forward it, its data memory becomes full. When this happens, the node has to drop some of the messages it is carrying. The decision regarding which messages to drop is a very important one, since dropping the last copy of a message may lead to the loss of that message, or dropping a copy of a message whose destination will be met in the near future can lead to a much higher delivery latency for that particular message. Recent socially-aware opportunistic routing algorithms such as BUBBLE Rap or Socio-Aware Overlay take advantage of nodes that have high social relationships with the other nodes in the opportunistic network. These nodes act as hubs, supporting most of the workload of the network, and can therefore easily become congested, especially if the number of such nodes in the network is not very large. The SAP algorithm attempts to solve this issue by only forwarding messages to nodes that have a high chance of encountering the destination in the near future, thus drastically reducing the total number of messages exchanged. BUBBLE Rap has the disadvantage that, when two nodes meet and one of them has a higher centrality than the other, the node with the lower centrality value will forward all its messages to the other node, regardless of destination, age or other factors. Another important difference between SAP and BUBBLE Rap is that, when a node's data memory is full, BUBBLE Rap drops the oldest message in its cache, whereas SAP drops the least important message in terms of utility value.

Several authors have treated the problem of network and node congestion in opportunistic networks. In [17], the authors propose a congestion-aware forwarding algorithm for opportunistic networks that aims to avoid the nodes and network areas that are congested, when performing routing. The algorithm is composed of three main parts, the first of them being a socially-aware component that selects the next hop of a message based on the probability

of it encountering the message's destination in the near future. This is similar to SAP, except that the socially-aware component described in [17] only uses social information to perform the prediction, and not the history of encounters. The second component is based on node resources, and attempts to detect and avoid the nodes that are congested or have a low availability, and the third component performs the same predictions, but for parts of the network instead of nodes. A set of heuristics for selecting the appropriate node are proposed by the authors, and four of them are compared to Epidemic, Prophet, Spray and Wait and Spray and Focus, and are shown to outperform all of them in terms of delivery latency, buffer availability and hit rate. In [2], the problem of congestion is solved through buffer management and flow control. Buffer management refers to the way data is dropped when there is a buffer overlow, and the authors propose eight dropping strategies: Least Interested, Most Interested, Max Copies, Most Forwarded, Least Forwarded, Random, Infinite Buffer and No Buffer. These strategies are similar to the one used by SAP, but they are simpler and only focus on a single characteristic. In our algorithm, we try to take into consideration various factors, such as the interest of other nodes, encounter prediction or age of a message. The flow control is used to prevent a sender from flooding a receiver with data, and this is performed by giving the receiver the capability of signaling its ability of handling incoming data. When the receiver's data memory is filling up, it notifies any sending nodes, and the flow of data is either stopped or slowed down. Testing shows that the Most Forwarded dropping strategy performs best in terms of hit rate, delivery latency and overhead. Another technique for limiting congestion is called storage routing [18] which, when a node is congested, uses neighboring nodes to migrate a set of messages to. However, we consider this to only be useful in networks with large contact durations and with many nodes. In sparse opportunistic networks, the lack of neighboring nodes at a given time might limit the usefulness of such a solution.

SAP performs congestion control by computing utilities for all the messages when a contact occurs (both the current node's messages, as well as the ones belonging to the encountered node). Based on these utilities, if a node's data memory doesn't have enough room for all the messages, only the ones with the highest utility values are kept. Therefore, by always maximizing the value of a node's data memory, we make sure that we drop the less important messages (i.e. the ones that the current node has a lower chance of successfully delivering to the destination).

V. Experimental Setup and Results

This section presents an experimental analysis of the SAP opportunistic routing algorithm in terms of node and network congestion in an academic environment. We focus on such an environment because we believe a university is a place where an opportunistic network can be implemented the easiest. However, we prove our results in other situations as well. We highlight the improvements SAP brings compared to distributed BUBBLE Rap [11]. The reason we chose BUBBLE Rap for comparison is that it is one of the most efficient opportunistic routing algorithms in terms of hit rate and delivery latency.

A. Setup

The experiments were performed on the UPB 2012 trace presented in Sect. III. We also tested on the St. Andrews trace because we wished to show that SAP behaves well in a different situation too, where nodes are not grouped together and the area of the trace encapsulates an entire town, not just a small enclosed space.

We modelled the generation of messages as closely as possible to a real-life academic situation. Therefore, every weekday each node generates 30 messages. The destinations are selected using a Zipf distribution with exponent 1, where the most messages are sent to nodes that are in the node's social network (i.e. the owners of the devices are Facebook friends). Inside a community, the nodes are chosen randomly. The time of day when messages are sent was selected after analyzing information regarding the traces and choosing the two-hour interval with the highest number of contacts. Messages are not sent at all in the final ten days of the experiment, since there wouldn't be enough time for them to reach their destinations due to the low number of participants. For the UPB 2012 trace, we eliminated the nodes that had less than 10 encounters for the entire duration of the trace, since the messages they would generate (or would be destined for them) would have very small chances of reaching their destinations.

There are two important parameters to the SAP algorithm: the data memory size and the cache size. The cache size represents the amount of encountered nodes that the algorithm remembers, and after empirical testing we decided to set this value to 40 (so a node remembers information about the last 40 encounters it had). The data memory size applies not only to SAP, but also to BUBBLE Rap and signifies the number of messages a node can carry. The main difference between SAP and BUBBLE Rap regarding the data memory is that, when a BUBBLE Rap node has the memory full and must receive another message, it drops the oldest message in its cache. On the other hand, SAP drops the message with the lowest utility value, i.e. the most useless message.

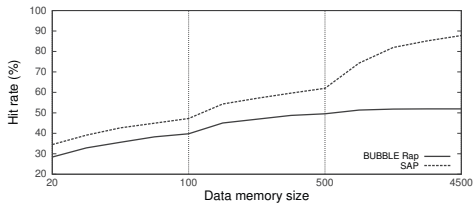
There are several metrics that we use for analyzing the simulation data. First of all, we look at hit rate, which is the ratio between data objects that have successfully arrived at requesting nodes and the total number of requests generated by all nodes. Although in this paper we treat the congestion problem, hit rate still remains the main goal of opportunistic routing algorithms. Achieving a hit rate close to 100% means that opportunistic networks are plausible for implementation

in real-life. Secondly, we consider the delivery cost as the ratio between the total number of messages exchanged during the course of the experiment and the number of generated messages. This is a measure of network congestion, since fewer messages sent in the network leads to a less congested network. Another measure of congestion, but this time at the nodes, is the hop count. It is computed as the number of nodes that carried a message until it reached the destination on the shortest path. Node congestion is also highlighted by the amount of buffer overflow events that occur during the duration of the experiment. A buffer overflow takes place when a node is receiving messages at a higher rate than it is able to forward, causing the data memory to fill up. We also analyze the saturation rate for each node, i.e. the number of overflow events a node experiences per time unit over the entire duration of the experiment.

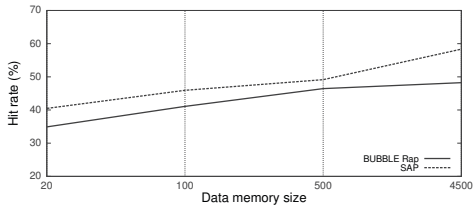
B. Results

Figure 1 shows a comparison between BUBBLE Rap and SAP for both traces in terms of hit rate, when varying the size of the data memory from 20 up to 4500. The maximum possible hit rate that can be achieved for UPB 2012 is 87.81% and we obtain it using SAP when a node can store 4500 messages. Although 4500 may be a large value, we can see from the figure that SAP performs very well even with smaller values (e.g. at 1500 we obtain 74.33%). Another important conclusion that can be drawn from Fig. 1 is that SAP outperforms BUBBLE Rap in terms of hit rate regardless of the size of the data memory. For small data memory values, the improvement of SAP is about 6-7%, but increasing the data memory size leads to as much as a 36% improvement brought by SAP. It can also be seen that the growth in hit rate with data memory size is linear for BUBBLE Rap, while for SAP it is close to an exponential function. The results are similar for the St. Andrews trace, where the hit rate for SAP is always better than for BUBBLE Rap. However, it never reaches the maximum possible hit rate (the maximum obtained with SAP is 58% out of 64%). Maximum hit rate probably could be achieved, but the data memory would have to be very large in order for that to be possible.

The delivery cost is shown in Fig. 2. It can be easily seen that BUBBLE Rap behaves very badly when the data memory size is increased (we have chosen to have a smaller Y on the figure so that the plot for SAP could be seen better, since the maximum delivery cost for BUBBLE RAP on the UPB 2012 trace is 6721, and only 224 for SAP). The reason for the bad behavior of BUBBLE Rap is that messages keep being moved from one node to another before eventually reaching the desired destination. SAP has no such problem and outperforms BUBBLE Rap even for small data memory sizes (for a data memory with 20 messages, BUBBLE Rap has a delivery cost of 101 on the UPB 2012 trace, while SAP has 9, meaning that only 9 times the total

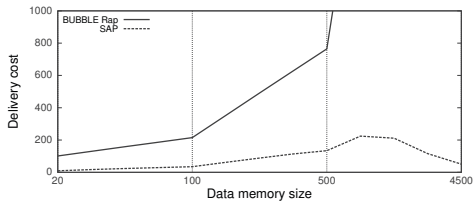


(a) UPB 2012

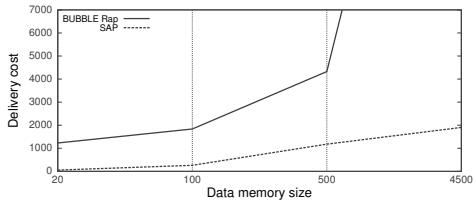


(b) St. Andrews

Figure 1. Hit rate for BUBBLE Rap and SAP.



(a) UPB 2012

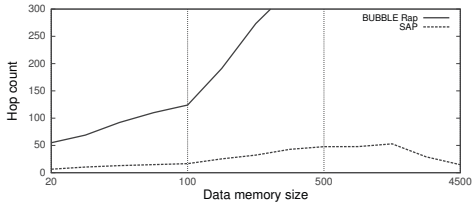


(b) St. Andrews

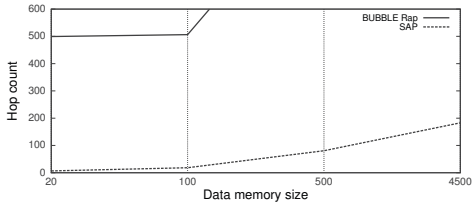
Figure 2. Delivery cost for BUBBLE Rap and SAP.

number of generated messages have been exchanged in the network). The St. Andrews results show the same behavior on a non-academic trace, which leads us to believe that the SAP algorithm behaves well for various types of traces. Having a low delivery cost means that there won't be too many messages being exchanged in the network at any given time, which leads to a low network congestion rate. When a new node enters the network, it can start sending messages without fear of them being dropped.

The situation regarding hop count is very similar to delivery cost, as shown in Fig. 3. SAP clearly outperforms BUBBLE Rap for both traces. We consider hop count to be a very important metric of opportunistic routing, since it directly influences bandwidth and node congestion. In Fig. 4 we show the total number of buffer overflow events that occurred for the entire duration of the tracing experiment at

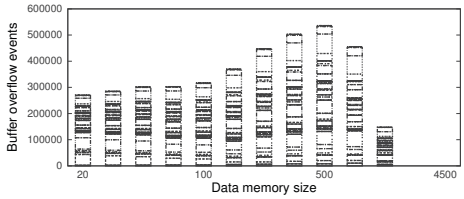


(a) UPB 2012

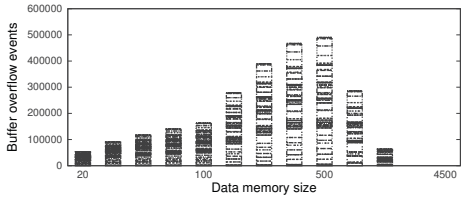


(b) St. Andrews

Figure 3. Hop count for BUBBLE Rap and SAP.

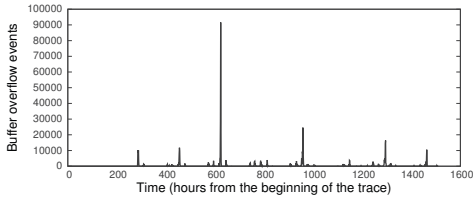


(a) BUBBLE Rap

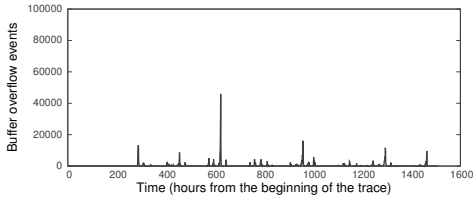


(b) SAP

Figure 4. Buffer overflow events for UPB 2012.



(a) BUBBLE Rap



(b) SAP

Figure 5. Saturation for UPB 2012.

UPB 2012. It can be seen that the number of overflow events at SAP is far smaller than the one at BUBBLE Rap, because messages are sent more rarely, as shown by the delivery cost and hop count results. For a data memory of more than 3500 messages, there are no buffer overflow events, since the size of the data memory is sufficient to store all necessary messages. However, this doesn't necessarily mean that the routing algorithm will yield optimal results, which can easily be seen by looking at the hit rates for BUBBLE Rap from Fig. 1. Sometimes, it is better to store more messages, if possible. It is also clear from Fig. 4 that the most overflow messages occur when the data memory can store a maximum of 500 messages, and generally all nodes (represented differently on the stacked histogram) encounter buffer overflow events at one point or another during the experiment. The output for St. Andrews is not showed here due to lack of space, but is very similar to UPB 2012.

Figure 5 shows the saturation rate for BUBBLE Rap and SAP on UPB 2012, computed hourly for a data memory of 500 messages, and it is clear that SAP has fewer overflow events, which was also shown in Fig. 4. However, an interesting conclusion regarding Fig. 5 is that there are spikes in the number of overflow events once in a while, which occur at the same time for both algorithms. After analyzing the trace, we found out that those spikes occur when a large number of nodes are together in the same place for a relatively long period of time. In such a situation, a node attempts to communicate with all other nodes in range, therefore a large number of messages is sent at once.

VI. Conclusion

We have presented in this paper an algorithm entitled Socially-Aware Prediction (SAP) that makes use of information about the social relationships between owners of the mobile devices from an opportunistic network when routing data. Moreover, the SAP algorithm can correctly predict the number of contacts a node will have in a given time interval. By combining this information, we were able to obtain a routing algorithm that can outperform an existing technique (distributed BUBBLE Rap) in terms of hit rate, delivery cost and hop count.

Furthermore, we have focused on the problem of congestion, which can appear when certain nodes with high values of social centrality receive more messages than they get a chance to forward. When this happens, the nodes' data memories become full and the nodes must drop some of the older messages. We showed that SAP chooses which messages to drop by applying a utility function to each message, and selecting the ones with the lowest value for dropping. Congestion is also limited by only sending data when there is a certain level of confidence that the

neighbor a node is forwarding to will encounter a message's destination in the near future. We showed experimentally that the number of overflow events and congested nodes in the network clearly decreases using SAP, both on an academic trace collected at our faculty, and on a different type of trace that was collected in a larger, sparser and more open environment.

ACKNOWLEDGMENT

This work was supported by project "ERRIC - Empowering Romanian Research on Intelligent Information Technologies/FP7-REGPOT-2010-1", ID: 264207. The work has been cofounded by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/89/1.5/S/62557. The paper has benefited from the collaborative research efforts of the EU Green-Net group.

REFERENCES

[1] Greg Bigwood, Devan Rehunathan, Martin Bateman, Tristan Henderson, and Saleem Bhatti. Exploiting self-reported social networks for routing in ubiquitous computing environments. In *Proceedings of the 2008 IEEE International Conference on Wireless & Mobile Computing, Networking & Communication*, pages 484–489, Washington, DC, USA, 2008. IEEE Computer Society.

[2] Fredrik Bjurefors, Per Gunningberg, Christian Rohner, and Sam Tavakoli. Congestion avoidance in a data-centric opportunistic network. In *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, ICN '11, pages 32–37, New York, NY, USA, 2011. ACM.

[3] Chiara Boldrini, Marco Conti, Franca Delmastro, and Andrea Passarella. Context- and social-aware middleware for opportunistic networks. *J. Netw. Comput. Appl.*, 33(5):525–541, 2010.

[4] Chiara Boldrini, Marco Conti, Jacopo Jacopini, and Andrea Passarella. HiBOp: a History Based Routing Protocol for Opportunistic Networks. In *World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE Int. Symp. on a*, pages 1–12, 2007.

[5] Chiara Boldrini, Marco Conti, and Andrea Passarella. Exploiting users' social relations to forward data in opportunistic networks: The HiBOp solution. *Pervasive Mob. Comput.*, 4:633–657, 2008.

[6] Chiara Boldrini, Marco Conti, and Andrea Passarella. Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks. *Computer Networks*, 54:589–604, March 2010.

[7] Radu Ciobanu and Ciprian Dobre. Data dissemination in opportunistic networks. In *18th Int. Conf. on Control Systems and Computer Science*, CSCS-18, pages 529–536, 2011.

[8] Radu I. Ciobanu and Ciprian Dobre. Predicting encounters in opportunistic networks. 2012.

[9] Radu Ioan Ciobanu, Ciprian Dobre, Valentin Cristea, and Dhiya Al-Jumeily. Social opportunistic networking. In *Proceedings of the 11th International Symposium on Parallel and Distributed Computing*, ISPDC 2012, 2012.

[10] Marco Conti, Silvia Giordano, Martin May, and Andrea Passarella. From opportunistic networks to opportunistic computing. *Comm. Mag.*, 48:126–139, 2010.

[11] Pan Hui, Jon Crowcroft, and Eiko Yoneki. BUBBLE Rap: social-based forwarding in delay tolerant networks. In *Proc. of the 9th ACM int. symp. on Mobile ad hoc networking and computing*, MobiHoc '08, pages 241–250, New York, USA, 2008. ACM.

[12] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Learning to detect events with markov-modulated poisson processes. *ACM Trans. Knowl. Discov. Data*, 1(3), December 2007.

[13] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 34(4):145–158, August 2004.

[14] Dmytro Karamshuk, Chiara Boldrini, Marco Conti, and Andrea Passarella. Human mobility models for opportunistic networks. *IEEE Comm. Magazine*, 49(12):157–165, 2011.

[15] Vincent Lenders, Martin May, Gunnar Karlsson, and Clemens Wacha. Wireless ad hoc podcasting. *SIGMOBILE Mob. Comput. Commun. Rev.*, 12:65–67, January 2008.

[16] Hoang Anh Nguyen, Silvia Giordano, and Alessandro Puiatti. Probabilistic Routing Protocol for Intermittently Connected Mobile Ad hoc Network (PROPICMAN). *2007 IEEE Int. Symp. on a World of Wireless Mobile and Multimedia Networks*, pages 1–6, 2007.

[17] Milena Radenkovic and Andrew Grundy. Congestion aware data dissemination in social opportunistic networks. In *Proceedings of the Eighth International Conference on Wireless On-Demand Network Systems and Services*, WONS, pages 60–67, 2011.

[18] Matthew Seligman, Kevin Fall, and Padma Mundur. Storage routing for dtn congestion control: Research articles. *Wirel. Commun. Mob. Comput.*, 7(10):1183–1196, December 2007.

[19] Chaoming Song, Yehui Qu, Nicholas Blumm, and Albert-László Barabási. Limits of Predictability in Human Mobility. *Science*, 327:1018–1021, 2010.

[20] Alan Stuart, Keith Ord, and Steven Arnold. *Kendall's Advanced Theory of Statistics, Classical Inference and the Linear Model*, volume Volume 2A (2007 reprint). Wiley, sixth edition, 1999.

[21] Eiko Yoneki, Pan Hui, ShuYan Chan, and Jon Crowcroft. A socio-aware overlay for publish/subscribe communication in delay tolerant networks. In *Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, MSWiM '07, pages 225–234, New York, NY, USA, 2007. ACM.