Leader Election in Opportunistic Networks

Radu Drăgan*, Radu-Ioan Ciobanu*, Ciprian Dobre*†
*University Politehnica of Bucharest, Bucharest, Romania
†National Institute for Research and Development in Informatics, Bucharest, Romania
Emails: radu.dragan@stud.acs.upb.ro, {radu.ciobanu, ciprian.dobre}@cs.pub.ro

***Abstract*—New technologies brought to life by the rapid growth in number of mobile devices are starting to be employed for creating cyberinfrastructures for smart cities. One such technology comes in the form of opportunistic networks (ONs), where mobile nodes only communicate between each other through protocols such as Bluetooth or Wi-Fi Direct. The usefulness of ONs in smart cities is high, because they are able to reduce the load on the city's infrastructure, while at the same time decreasing the response time. Multiple opportunistic nodes need to collaborate towards opportunistic computing and smart cyberinfrastructures, and for this reason we argue that the problem of consensus in ONs is of the utmost importance. Our aim is to propose a general-purpose consensus algorithm for opportunistic networks. However, to expect a given portion of all the nodes inside an opportunistic networks to agree upon a subject is certainly overly-optimistic. Thereby, we aim to group the nodes in communities based on their closeness to each other and the time they spend together. Then, a leader is elected in each community, and the leader will be in charge with the consensus decision. In this paper, we focus on grouping nodes into communities and electing the leader in each community. We propose and analyze two approaches, which we compare in order to find the most suitable one.**

***Keywords*-opportunistic; networking; leader; consensus;**

I. Introduction

In recent years, the number of mobile devices of all shapes and sizes (ranging from small sensors to powerful smartphones or laptops) has increased constantly. In 2016, the number of unique mobile users was 4 billion [1], so more than half of the Earth's population owns a mobile device, from only 20% 10 years ago. By 2020, it is estimated that another new billion additional mobile subscribers will appear, taking the global rate to 60%. In these conditions, the opportunistic networking paradigm has come to the forefront of researchers in the area of mobile networks and smart cities alike. Opportunistic networks (ONs) are generally formed only of mobile devices, without needing any external infrastructure. Nodes only communicate between each other when they are in close range, through protocols such as Bluetooth or Wi-Fi Direct. This has the potential to reduce the load of mobile traffic from the Wi-Fi access points and broadband connections towards the nodes themselves, while at the same time offering faster speeds and a more efficient communication for the end-users.

There are many real-life scenarios where opportunistic networks are being (or have already been) employed, such as disaster management [2], smart cities [3], floating content [4], mobile advertising [5], crowd management [6], context-aware platforms [7], wildlife tracking [8], Internet access in limited conditions [9], distributed social networks [10], or even data offloading and mobile cloud computing [11]. In this paper, we choose to focus on implementing a cyberinfrastructure for smart cities. ONs are extremely useful in smart cities, since they not only decrease the load on the infrastructure, but they also improve the response times. Instead of requiring all nodes in the smart city (such as simple sensors, smartphones, wireless routers, etc.) to be connected to the Internet, opportunistic networks allow the nodes to communicate between each other, and only some specialized nodes to send information to some processing servers or the cloud. These nodes can opportunistically collect relevant data from other nodes, aggregate them, and only then connect to the Internet to send them.

Aside from reducing the stress on the Internet infrastructures, opportunistic networks can also improve the security of smart cities. Instead of sending everything to the cloud, which may or may not be secure, data that only have local relevance can be processed close to where they are generated. For example, a smart thermostat does not need to connect to a server to decide that the data read from some temperature sensors suggest that the room temperature should be increased.

In such situations, multiple opportunistic nodes from a smart city need to work together with a common goal, not only regarding communication, but also in terms of behavior. To highlight the need for opportunistic consensus, we propose the following scenario. Users traveling in the smart city by car have an application installed on their phones which they can use to report various traffic incidents such as crowded streets, gridlocks, accidents, bad roads, construction works, etc. Normally, the application would collect this information from the users at a central server (or in a cloud), process it, and show the users a view of the entire city (which is similar to what Waze[1] does). However, let us assume that the goal is to reduce the costs required for infrastructure (in this situation, cloud time for running the app servers), while also allowing users with no Internet connection to use the application properly. In this case, opportunistic networking can be used to perform communication between devices located in the same geographical area, which would exchange data between themselves, offering users in the area a partial view of the smart city (i.e., the part which is of interest to

[1] https://www.waze.com.

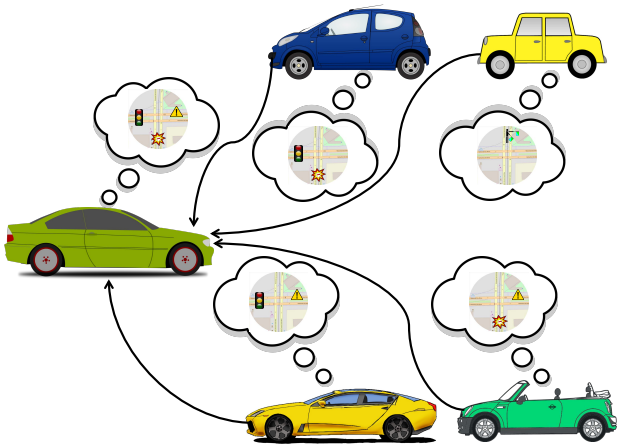


Figure 1. Smart city scenario for consensus in opportunistic networks.

them). Whenever a user wants to report something through the app, that information is spread to all encountered nodes on a certain radius, which in turn pass it on to other peers they encounter, and so on. In our previous work [12], we proposed an algorithm for disseminating multimedia content in ONs, which would prove useful in situations when the drivers would want to share photos or videos regarding the traffic. However, the important question here is how to decide if an incident reported by a user is true, and this is where consensus comes into play. Namely, it allows multiple nodes to corroborate their information, and decide whether it is correct or not. This scenario is presented in Fig. 1, where the green car on the left receives traffic information from the other four cars. However, since the incident signaled by the yellow car in the top right corner is not mentioned by another car, no consensus can be reached, so the event is not registered by the green car.

There are very few solutions that focus on issues beside simple ON communication, and the problem of consensus is very lightly addressed. Still, we believe that, since there is no central entity and ONs are fully decentralized, there can always be malicious nodes that cannot be trusted, and these nodes should be eliminated not only through trust and reputation algorithms, but also through solutions that are able to reach consensus between a large number of nodes that have no other means to communicate with each other except through close-range protocols.

Our aim is to propose a general-purpose consensus algorithm for opportunistic networks. However, since it is highly unlikely that most of the nodes inside an opportunistic networks would agree upon a subject, we aim to group the nodes in communities based on their closeness to each other and the time they spend together. Each community then elects a leader, which is in charge of the consensus decision. It receives opinions from all nodes of the community, and based on these opinions it makes decisions which are then disseminated to all the nodes. In this paper, we focus on clustering nodes into communities and electing the leader in each community. We thus propose two different approaches, which we compare in order to find the most suitable one. The first approach implies that each node disseminates a candidacy, which is then used by the receiving nodes for computing a score. The node with the higher score becomes the leader of the current node. The second approach relies on community detection, each community having the task to reach a consensus regarding its leader.

The rest of the paper is structured as follows. In Sect. II, we present related work in the area of consensus finding, both in distributed systems, as well as in opportunistic networks in particular, while also analyzing solutions for leader election. Then, we propose two methods of our own for choosing the leaders in ONs in Sect. III. Section IV presents an analysis on the behavior of our proposed solutions, while finally Sect. V highlights our conclusions and future work.

II. Related Work

The subject of finding consensus in opportunistic networks has been gaining some importance recently, given the fact that some situations when the nodes would have to agree upon a subject might arise quire often.

Probably the most well-known algorithm regarding consensus in distributed systems is Paxos [13], [14], and was used for systems like the lock service of Google File System (GFS), Google Chubby [15], Apache Zookeper [16], etc. Most of these solutions may consider node failures, which are common in every distributed system, but they generally do not take into account link failures, which are more than common in opportunistic networks. Nodes can only communicate directly as long as they are in wireless range of each other, so the contact time is really low and links appear and disappear in an instant from the network. Opportunistic network solutions do not see this as an issue, but rather as the expected behavior of the network, so this should not be considered a corner case. This is one of the most important matters that any consensus solutions for ONs should deal with.

The first important approach that aims to consider both kinds of failures is the Heard-Of model [17], and this could be one of the starting algorithms for reaching consensus in our type of network. An algorithm based on the Heard-Of model is the “One Third Rule”, which needs a number of rounds to reach consensus. At each round, a node disseminates its proposed values to the other nodes. Upon receiving a given fraction of the other nodes’ opinions, each node computes the most frequent value received and that will become the new proposal. When a node receives a preset number of opinions with the same value, it selects that value. After making a decision, the node disseminates it to the other nodes which have to comply, and so consensus is reached. The approach was used in [18], where the authors implemented the algorithm on a small fleet of devices, justifying that the approach would fit the demands of running an opportunistic network in a real-life scenario.

Another way of reaching consensus is by making use of a leader such as in [19]. The algorithm assumes

that a node is already elected, without giving any detail regarding this leader election process. Then, the leader receives proposals from the other nodes. When the leader receives proposals from the majority of nodes, it decides and communicates this decision to all the nodes from its community. After receiving the decision, each node sends back an acknowledgement to the leader. When receiving the expected number of acknowledgements, the leader validates the decision and sends the validation to the other nodes. At this point, the consensus is reached.

The issue with the first approach in opportunistic networks (especially dense ones, with many contacts) is that it can lead to congestion, because a large amount of messages need to be exchanged between all the nodes. Furthermore, if the communities are not tight, the number of rounds required to achieve consensus can be extremely large, so it would take much time until all nodes agree. Thus, in our opinion, electing a leader would be more suitable for such networks, and this is what we focus on in this paper.

Although the leader is considered known in [19], electing it is not a simple problem, which is why, in this paper, we attempt to select per-community leaders, and allow them to act on behalf of their communities in the consensus solution. While not explicitly aiming to choose a leader for reaching consensus, there have been several solutions over the years that split the opportunistic network in groups (communities) and select nodes with certain properties that act as leaders. In BUBBLE Rap [20], nodes only spread messages to peers that are more popular than themselves, using the betweenness centrality, which is the number of times a node is on the shortest path between two other nodes in the network. Similarly, the Socio-Aware Overlay algorithm [21] is a data dissemination technique that creates an overlay for an opportunistic network with publish/subscribe communication. The overlay is composed of nodes having high centrality values that have the best visibility in a community. These nodes act as brokers for a publish/subscribe paradigm, storing information about subscriptions and directing requests in the network. Other similar social-based solutions (such as ML-SOR [22] and SRSN [23]) use online social information (obtained from social networks such as Facebook) when grouping nodes together in opportunistic networks.

III. Leader Election

Moving forward, we focus on leader election in opportunistic networks, which is the first step in implementing consensus solutions. Initially, we attempted to reach a point where a single leader is elected in the entire opportunistic network. Each node would disseminate its centrality, which would act as a candidacy. Then, all nodes that received a centrality would compute a score by multiplying the centrality of the candidate with the trust that the current node had in the candidate. When a score higher than the current one was obtained, the candidate would become the new leader. We tested on one mobility trace and one synthetic scenario. The trace, called Sigcomm [24], was collected at a conference held in Barcelona, and involved 76 users for 4 days. We chose it because it offers social network information about the users, along with opportunistic contacts. Furthermore, it was captured in a scenario that has a high applicability for ONs.

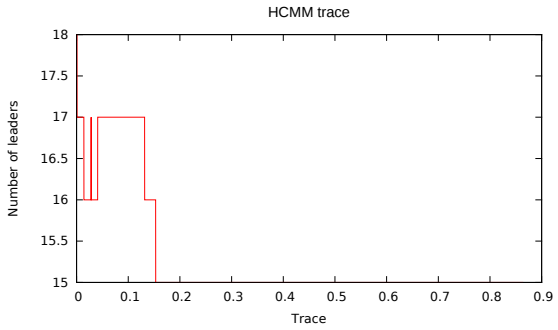
The mobility model used was HCMM (the Home-Cell Community-based Mobility Model) [25], a synthetic mobility model that tries to replicate the behavior of nodes in a mobile network. It is based on the caveman model [26], which assumes that nodes have home communities, where they spend most of their time, and acquainted communities, which they visit more rarely. Nodes in HCMM are driven not only by the social relationships between them, but also by the attractions of physical locations. According to this model, the attraction of an external cell is computed based on the relationships with nodes that have their home in that cell. When testing with HCMM, we tried to simulate a dense network split into nodes grouped into communities, with many contacts between each other. Such a scenario is meant to simulate a busy intersection, where many cars pass in all direction and exchange information often (similar to the use case presented in Sect. I). Therefore, we tested with a physical space represented by a 1000x1000-meter grid, with 10x10-meter cells. The speed of the nodes was chosen between 1.25 and 1.5 meters per second, while the transmission radius of the nodes was 10 meters, which is the regular Bluetooth range. There were 40 nodes in the network, split into four communities. The duration of the experiment was two hours (leading to more than 2300 contacts), and we used the HCMM community grouping to create the social network used for routing decisions.

However, this method did not produce the desired results. After the entire trace was completed, the number of elected leaders was 15 for the HCMM trace and 11 for the Sigcomm trace. The way results were obtained is detailed in Sect. IV. The evolution in time of the number of leaders throughout each trace is presented in Fig. 2. Thinking optimistically, we could assume that there is a central and trustworthy enough node so that it will be considered leader by all the other nodes, if these nodes receive its candidacy. However, the high mobility of nodes and the sometimes high delay of message transmissions make this consensus hard to reach. Employing a single leader for an entire opportunistic network could also bring up some real performance issues, because, however central a node, it will not be easily accessible by all the nodes of the network.

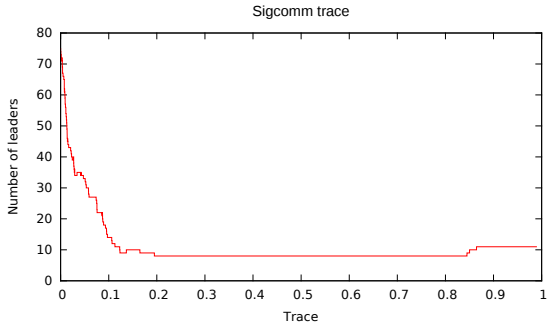
For this reason, we propose two approaches of electing a leader inside a community. Then, these leaders could cooperate in finding a master leader that could supervise all the other leaders. The following two subsections present our methods in detail.

A. Direct Approach

The first approach, presented in Fig. 3, implies that each node disseminates its centrality, similarly to the way



(a) HCMM



(b) Sigcomm

Figure 2. Evolution of the number of leaders in time.

BUBBLE Rap performs routing in opportunistic networks. An important characteristic of a leader is that it should be easily accessible to the nodes in its group, and centrality is the measure of this property. A central leader generally has a large number of contacts, so it is easily accessible. Each node that receives a centrality upon a contact (which acts as a candidacy) computes a leader score. We set the maximum hop count for the candidacy to a given number *d*, so the initiator of the candidacy is in close proximity. We recommend *d* to be set to small values (like 2 or 3), so the nodes gathered around a leader are relatively close to each other. When finding a node with a better score than the current one (i.e., a better leader for the current node), the leader seen by the current node changes, so the current node has to send a message to the old leader notifying it about the change, and to the new leader requesting access to its community. The new leader then sends a confirmation response back to the node and includes it in its community. This way, only the leader keeps track of the communities, a community being formed out of all the nodes that share the same leader. The leader score is computed by using the following formula:

$$score = w_t \times v_t + w_c \times v_c + w_p \times v_p + w_l \times v_l \quad (1)$$

The notations of the elements are as follows: w_t, w_c, w_p, and w_l are weight values for trust, centrality, contact probability, and latency, while v_t, v_c, v_p, and v_l are values for the aforementioned properties of a node. In consequence, we use the trust value of a node, which is computed as in [27], based on the number of successful

$leader = A$
$leader\_score = score(A)$
if contact with node B **then**
  $new\_cand = B.candidacies - A.candidacies$
  for all candidacies C in new_cand **do**
    $candidate\_score = score(C.node\_id)$
    if $candidate\_score > leader\_score$ **then**
      $new\_leader = C.node\_id$
      $leader.leave\_community(A)$
      $new\_leader.join\_community(A)$
      $leader = new\_leader$
      $leader\_score = candidate\_score$
    end if
  end for
end if

Figure 3. Direct approach for leader election. The algorithm is presented from the standpoint of node A.

message deliveries by a node. We chose to employ trust because we believe that a leader should be, first of all, trustworthy and cooperate with the other nodes in finding consensus.

For the centrality metric, we employed the single window (S-window) algorithm, which computes centrality by counting the encounters the current node has had in the last time window (set to six hours), and then performing an exponential smoothing on the cumulated values.

The method of computing the probability was successfully used by us in [28], and is based on observations made in [29] that, since generally opportunistic nodes are devices such as smartphones carried by humans, their mobility follows some patterns and could be predicted with a given accuracy. The probability used in the leader score computation is based on the history of contacts and on social information.

The last value used (v_l) is the latency of the candidacy, computed as the time between the moment when the candidacy was generated, and the time it was received by the node. This value is important because we want to make sure the requests from nodes to the leaders arrive as soon as possible, and the nodes can count on a relatively quick response from the leader. If the node does not receive it in a timely manner, it is ignored.

B. Community-Based Approach

The second approach, presented in Fig. 4, assumes that the communities are already established when the election of the leader begins. When two nodes spend more than a given time in contact, they include each other in their local communities. Each time a node A includes a node B in its local community, A checks if it can include B in its leader community (i.e., the community which would have to agree upon a leader). Therefore, node A asks the other nodes from its leader community if it can include B, by sending a request to all the nodes. When receiving a request, a node checks if B is in its local community or in its online social community (taken from

```
leader = A
leader_score = score(A)
leader_community = {A}
local_community = {A}
online_community = get_social_info()
if B should be added to local_community then
  count = 0
  for all nodes N in leader_community do
    if N.local_community.contains(B) then
      count++
    else if N.online_community.contains(B) then
      count++
    end if
  end for
  if count > t1 then
    leader_community.add(B)
    for all nodes N in leader_community do
      N.leader_community.add(B)
    end for
    if score(B) > leader_score then
      count = 0
      for all nodes N in leader_community do
        if score(B) > N.leader_score then
          count++
        end if
      end for
      if count > t2 then
        leader = B
        leader_score = score(B)
        for all nodes N in leader_community do
          N.leader = B
          N.leader_score = score(B)
        end for
      end if
    end if
  end if
end if
```

Figure 4. Community-based approach for leader election. The algorithm is presented from the standpoint of node A.

platforms such as Facebook, Twitter, LinkedIn, etc.). If this is the case, the node responds positively, otherwise it responds negatively. When the initiator of the request (A) receives confirmations from a given fraction t_1 of the community nodes (which could be dynamically set to satisfy the demands of the given structure of the network), A decides that B can be included in the leader community and thus disseminates this decision to all the nodes. After receiving a decision, the other nodes also include B in their leader community, so that all the nodes that take part in a leader community have a consistent view of it.

When being added to a leader community, node B becomes a candidate for the leader position. If one of the nodes computes a leader score for B higher than the one for the current leader, it asks the other nodes from the community about this matter. If over a given portion of nodes t_2 (usually half to form a majority) respond positively, then B is appointed as leader in the community, and the node that initiated the request also disseminates the decision, so that all the nodes from the community are aware of it. There is also a separate case that we had to consider. Some nodes move away from their community neighbors. Thus, from time to time, a node forms two sets: one with community nodes seen in the last time period, and one with nodes not seen. Then, the current node formulates remove requests for all the nodes it has not seen, and forward them to all the nodes seen. When a node receives at least a given fraction of remove requests regarding a node, and it did not meet that node in the last time interval, it removes the node from its community and disseminates this decision to the seen nodes, which have to comply.

For this approach, the formula is similar to Equation 1, the only difference being the absence of the latency value (and corresponding weight), because there are no more leader candidacies:

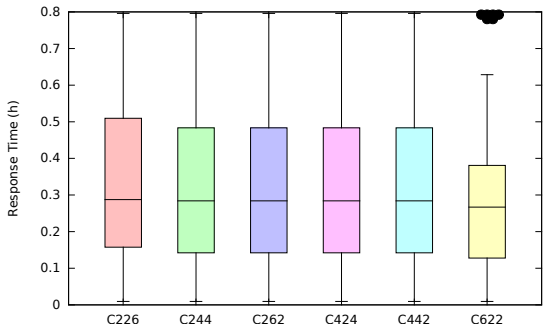
$$score = w_t \times v_t + w_c \times v_c + w_p \times v_p \qquad (2)$$

One of the advantages of the direct approach is its simplicity, and the fact that the nodes are grouping around a leader, which would mean that all nodes inside the same community have an easy access to the leader. Regarding the community-based approach, the communities are tighter, given the stricter algorithm used to form them. This way, the nodes in a group should be closer to each other, and the addition and removal of a node and election of a leader should be performed rapidly.

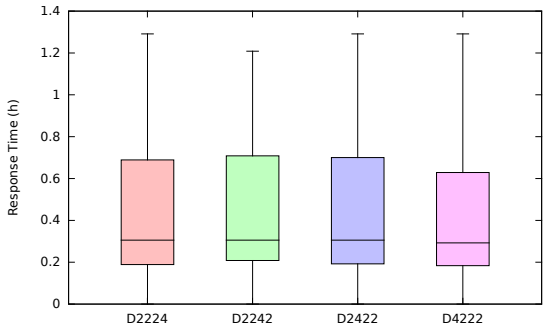
IV. Experimental Results

We used the MobEmu[2] Java-based simulator to implement and test the two algorithms. MobEmu is a simulator able to replicate the mobility of the nodes by using data from synthetic and real-life traces, which record the duration and frequency of contacts between nodes and, possibly, the social relationships between them. We tested the algorithms on the two traces described in Sect. III, namely HCMM and Sigcomm. Each node sends from time to time heartbeat messages to its leader, waits for the response and then computes the response time in hours. We consider this measure relevant to our goals, given that the leader should be able to respond quickly to other nodes' requests. The results in terms of response times are provided in Fig. 5 and Fig. 6. We varied the weight values so that they are multiples of 0.2 and their sum is equal to 1, considering all the possibilities that respected these constraints. Regarding the notation presented in the figures, D2224 would mean running the direct approach with weight values of 0.2, 0.2, 0.2, and 0.4 for trust weight, centrality weight, probability weight, and latency weight, respectively. C226 would mean running the community-based approach with weight values of 0.2, 0.2, 0.6 for trust, centrality, and probability weight.

[2]https://github.com/raduciobanu/mobemu.

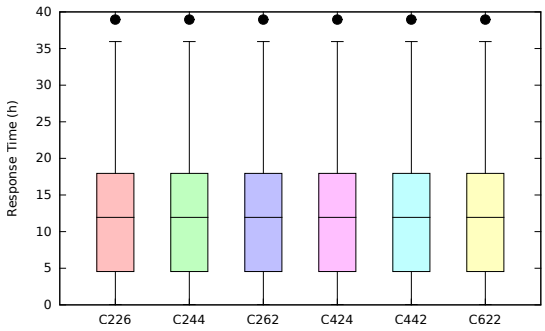


(a) Community-based approach

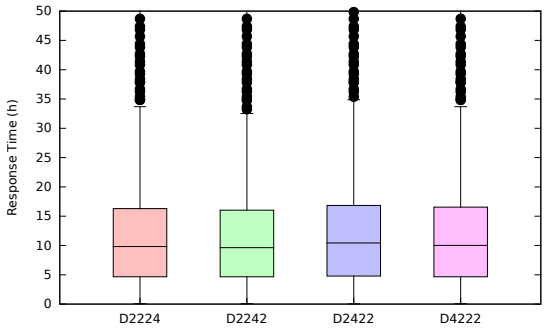


(b) Direct approach

Figure 5. Response times for the HCMM trace.



(a) Community-based approach



(b) Direct approach

Figure 6. Response times for the Sigcomm trace.

Fig. 5 highlights the results obtained by running the algorithms for the HCMM trace, where we simulated a crowded intersection environment, having an average of almost 29 contacts per hour for each node. As it can be observed, the response times for the HCMM trace have an average under 18 minutes (30% of an hour), which is really promising considering that we are dealing with a Delay-Tolerant Network. Furthermore, it can be observed that the results for the two approaches are similar. It can also be seen in Fig. 5 that varying the weights for the score function does not really affect the outcome of either approach. This happens because nodes that are fit to be leaders generally have high values for all the components of the score function (i.e., they are popular and thus have a high chance of encountering many nodes, they are trustworthy because they deliver a lot of messages, etc.). The only anomaly can be observed for the C622 test case, where we can see several outliers in Fig. 5, so a conclusion that can be drawn here is that the trust component should not be assigned a very high value, because some nodes that are trustworthy are not necessarily fit to be leaders (without taking into account the other score function parameters).

The results for the Sigcomm trace are shown in Fig. 6 and make us really optimistic regarding running the two approaches in scenarios other than vehicular ones. For the community-based approach, the average response time is 12 hours, while for the direct approach the average is around 10 hours. The direct-based solution behaves a little better considering that the nodes tend to group around leaders and also the overhead regarding the maintenance of the communities is skipped. Also the communities should be more dynamic for the direct approach. Again, the weight combinations do not seem to affect the outcome much, but it can be observed that the best values for the direct approach seem to be obtained when the centrality has the highest weight.

We also computed a ratio between the number of contacts between two random nodes from the same community, and a node and its leader, in order to highlight the benefit of having a leader when trying to disseminate a message or a decision towards the other members of a node's community. We obtained a value of 69% for the HCMM trace and 79% for the Sigcomm trace. This analysis proves that a node will have more contacts with his leader than with a random node from his community. This ratio was computed only for the community-based approach, because in the direct approach only the leader knows the members of its community.

V. Conclusions and Future Work

In this paper, we introduced two approaches of electing a leader in opportunistic networks, with the goal of employing them in the future for implementing consensus algorithms in ONs. These would be useful in many situations where opportunistic networks are employed, like smart city infrastructures for vehicular data dissemination (such as multimedia sharing). We compared our two proposed solutions in terms of the specifics of opportunistic networks, testing them on two real-life mobility traces. We analyzed the results in terms of response times, obtaining

some promising results towards implementing consensus algorithms.

Despite the fact that the results for response times are satisfactory, there is still room for improvement. The algorithms should be tested against a wider variety of traces, and they could definitely benefit from a mechanism that would enable them to adapt to the structure of the network. Once we obtain an efficient leader election, we will continue by proposing our consensus solution for opportunistic networks.

ACKNOWLEDGMENT

The research presented in this paper is supported by projects: DataWay (PN-II-RU-TE-2014-4-2731), MobiWay (PN-II-PT-PCCA-2013-4-0321), and Traffic and Data Offloading in Mobile Networks - TTOff, H2020 as part of Measuring Mobile Broadband Networks in Europe.

We would like to thank the reviewers for their time and expertise, constructive comments and valuable insight.

REFERENCES

[1] GSMA, "The mobile economy," http://www.gsma.com/mobileeconomy/, 2016, [Online; accessed 07-February-2017].

[2] L. Lilien, A. Gupta, and Z. Yang, "Opportunistic networks for emergency applications and their standard implementation framework," in *Proc. of IEEE International Performance, Computing, and Communications Conference*, ser. IPCCC 2007, 2007, pp. 588–593.

[3] V.-D. Le, H. Scholten, and P. Havinga, "Unified routing for data dissemination in smart city networks," in *Proc. of the 3rd International Conference on the Internet of Things*, ser. IOT 2012. USA: IEEE Press, 2012, pp. 175–182. [Online]. Available: http://doc.utwente.nl/83395/

[4] M. S. Desta, E. Hyytiä, J. Ott, and J. Kangasharju, "Characterizing content sharing properties for mobile users in open city squares," in *Proc. of the 10th Annual Conference on Wireless On-demand Network Systems and Services*, ser. WONS 2013, 2013, pp. 147–154.

[5] A. Heinemann and T. Straub, "Opportunistic networks as an enabling technology for mobile word-of-mouth advertising," in *Handbook of Research on Mobile Marketing Manag.*, ser. PA: Business Scrience Reference, K. Pousttchi and D. G. Wiedmann, Eds. Hershey, 2010, pp. 236–254.

[6] I. Wakeman, S. Naicken, J. Rimmer, D. Chalmers, and C. Fisher, "The fans united will always be connected: building a practical dtn in a football stadium," in *ADHOCNETS 2013, 5th International Conference on Ad Hoc Networks*, Barcelona, Spain, October 2013.

[7] C. Dobre, F. Manea, and V. Cristea, "CAPIM: A context-aware platform using integrated mobile services," in *Proceedings of IEEE International Conference on Intelligent Computer Communication and Processing*, ser. ICCP 2011, 2011, pp. 533–540.

[8] T. Small and Z. J. Haas, "The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way)," in *Proc. of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '03. New York, NY, USA: ACM, 2003, pp. 233–244. [Online]. Available: http://doi.acm.org/10.1145/778415.778443

[9] S. Guo, M. Derakhshani, M. H. Falaki, U. Ismail, R. Luk, E. A. Oliver, S. U. Rahman, A. Seth, M. A. Zaharia, and S. Keshav, "Design and implementation of the KioskNet system," *Computer Networks Journal*, vol. 55, no. 1, pp. 264–281, Jan. 2011. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2010.08.001

[10] K. Thilakarathna, A. C. Viana, A. Seneviratne, and H. Petander, "The power of hood friendship for opportunistic content dissemination in mobile social networks," INRIA, Saclay, France, Tech. Rep., 2012.

[11] R.-C. Marin, "Hybrid contextual cloud in ubiquitous platforms comprising of smartphones," *Int. J. Intell. Syst. Technol. Appl.*, vol. 12, no. 1, pp. 4–17, Jul. 2013. [Online]. Available: http://dx.doi.org/10.1504/IJISTA.2013.055101

[12] R. Dragan, R.-I. Ciobanu, C. Dobre, C. X. Mavromoustakis, and G. Mastorakis, "Multimedia sharing over opportunistic networks," in *Intelligent Computer Communication and Processing (ICCP), 2016 IEEE 12th International Conference on*. IEEE, 2016, pp. 409–416.

[13] L. Lamport, "The part-time parliament," *ACM Trans. Comput. Syst.*, vol. 16, no. 2, pp. 133–169, May 1998. [Online]. Available: http://doi.acm.org/10.1145/279227.279229

[14] A. Ailijiang, A. Charapko, and M. Demirbas, "Consensus in the cloud: Paxos systems demystified," in *2016 25th International Conference on Computer Communication and Networks*, ser. ICCCN. IEEE, 2016.

[15] M. Burrows, "The chubby lock service for loosely-coupled distributed systems," in *Proc. of the 7th Symposium on Operating Systems Design and Implementation*, ser. OSDI '06. Berkeley, CA, USA: USENIX Association, 2006, pp. 335–350. [Online]. Available: http://dl.acm.org/citation.cfm?id=1298455.1298487

[16] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed, "Zookeeper: Wait-free coordination for internet-scale systems," in *Proc. of the 2010 USENIX Conference on USENIX Annual Technical Conference*, ser. USENIXATC'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 11–11. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855840.1855851

[17] B. Charron-Bost and A. Schiper, "The Heard-Of model: computing in distributed systems with benign faults," *Distributed Computing*, vol. 22, no. 1, pp. 49–71, 2009. [Online]. Available: http://dx.doi.org/10.1007/s00446-009-0084-6

[18] A. Benchi, P. Launay, and F. Guidec, "Solving consensus in opportunistic networks," in *Proc. of the 2015 International Conference on Distributed Computing and Networking*, ser. ICDCN '15. New York, NY, USA: ACM, 2015, pp. 1:1–1:10. [Online]. Available: http://doi.acm.org/10.1145/2684464.2684479

[19] F. Borran, R. Prakash, and A. Schiper, "Extending paxos/lastvoting with an adequate communication layer for wireless ad hoc networks," in *Proc. of the 2008 Symposium on Reliable Distributed Systems*, ser. SRDS '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 227–236. [Online]. Available: http://dx.doi.org/10.1109/SRDS.2008.21

[20] P. Hui, J. Crowcroft, and E. Yoneki, "BUBBLE Rap: social-based forwarding in delay tolerant networks," in *Proc. of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. MobiHoc '08. New York, USA: ACM, 2008, pp. 241–250. [Online]. Available: http://doi.acm.org/10.1145/1374618.1374652

[21] E. Yoneki, P. Hui, S. Chan, and J. Crowcroft, "A socio-aware overlay for publish/subscribe communication in delay tolerant networks," in *Proc. of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, ser. MSWiM '07. New York, NY, USA: ACM, 2007, pp. 225–234. [Online]. Available: http://doi.acm.org/10.1145/1298126.1298166

[22] A. Socievole, E. Yoneki, F. De Rango, and J. Crowcroft, "Opportunistic message routing using multi-layer social networks," in *Proc. 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

[23] G. Bigwood, D. Rehunathan, M. Bateman, T. Henderson, and S. Bhatti, "Exploiting self-reported social networks for routing in ubiquitous computing environments," in *Proc. of the 2008 IEEE International Conference on Wireless & Mobile Computing, Networking & Communication*, ser. WIMOB '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 484–489. [Online]. Available: http://dx.doi.org/10.1109/WiMob.2008.86

[24] A.-K. Pietiläinen, E. Oliver, J. LeBrun, G. Varghese, and C. Diot, "Mobiclique: Middleware for mobile social networking," in *Proc. 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

[25] C. Boldrini and A. Passarella, "HCMM: Modelling spatial and temporal properties of human mobility driven by users' social relationships," *Comput. Commun.*, vol. 33, pp. 1056–1074, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2010.01.013

[26] J. Wu and D. J. Watts, "Small worlds: the dynamics of networks between order and randomness," *SIGMOD Rec.*, vol. 31, no. 4, pp. 74–75, Dec. 2002. [Online]. Available: http://doi.acm.org/10.1145/637411.637426

[27] R.-I. Ciobanu, C. Dobre, M. Dascălu, S. Trăuşan-Matu, and V. Cristea, "SENSE: A collaborative selfish node detection and incentive mechanism for opportunistic networks," *Journal of Network and Computer Applications*, vol. 41, pp. 240–249, May 2014. [Online]. Available: http://dx.doi.org/10.1016/j.jnca.2014.01.009

[28] R. I. Ciobanu, C. Dobre, and V. Cristea, "SPRINT: Social prediction-based opportunistic routing," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*. IEEE, Jun. 2013, pp. 1–7. [Online]. Available: http://dx.doi.org/10.1109/wowmom.2013.6583442

[29] ——, *Social Aspects to Support Opportunistic Networks in an Academic Environment*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 69–82. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31638-8_6