Field Experiment on the Performance of an Android-Based Opportunistic Network

Andre Ippisch[(✉)], Philipp Brühn, and Kalman Graffi

Heinrich-Heine-University Düsseldorf,
Universitätsstraße 1, 40225 Düsseldorf, Germany
{ippisch,philipp.bruehn,graffi}@hhu.de

Abstract. Android smartphones ubiquitously available, they are mobile and have sophisticated communication opportunities. With Opportunistic Networks, we can use the wireless connectivity of smartphones and other smart devices to relay messages in store-carry-forward fashion from one node to another to implement novel data-oriented applications. We can use these networks for high-bandwidth local data transfers, in cases with low or no connectivity, such as in third-world countries or remote areas, or in cases where communication should not leave any traces. In the last years, we developed an Android application for Opportunistic Networking, named *opptain*, that can be deployed on off-the-shelf unrooted smartphones and smart devices, enabling to harness this idea by simply installing an app. As the quality of such networks is essential, we implemented a test framework for Android-based opportunistic networks to run tests and aggregate results automatically. In this paper, we present the evaluation results of a field experiment we conducted with the *opptain* application, in which we used 26 devices to evaluate the outcome typical use cases. The tests show that the expected quality is reached and provides robust performance for various applications. In total, *opptain*, the testing environment, as well as the results themselves, are promising; for an office scenario in which interference is more common than in other possible scenarios, we achieved encouraging results.

Keywords: Opportunistic Networks · Android · Smartphones
Smart devices · Field tests · Measurement study

1 Introduction

Opportunistic Networks (OppNets) are disorganized Delay Tolerant Networks (DTNs) with typically not existing end-to-end paths between nodes at a given time. Nodes in OppNets can be represented by, among others, human-carried equipment like smartphones, tablets, and other smart devices. Since there is no end-to-end path between nodes, *Store, Carry and Forward* routing is used in these networks. By this, messages can be passed on from node to node in

© Springer Nature Switzerland AG 2019
G. Mencagli et al. (Eds.): Euro-Par 2018 Workshops, LNCS 11339, pp. 547–558, 2019.
https://doi.org/10.1007/978-3-030-10549-5_43

the proximity, whenever an opportunity of data exchange occurs, which happens through the mobility of the nodes. We can classify routing schemes into flooding-based and utility-based approaches. The simple Epidemic [10] routing approach is a flooding-based routing scheme which replicates messages to every encountered node that does not yet own a message copy. One of the utility-based routing schemes, on the other hand, is PRoPHET [5] which relays messages only if the connected node has higher delivery predictability.

Researchers working on OppNets are only able to simulate sufficiently large networks. For simulating OppNets, there are multiple tools available [1], such as *The ONE Simulator* [4] and *PeerfactSim.KOM* [2]. We can take results from the simulators to improve real-life OppNets, but it is also desirable that results of real-life OppNets improve the parameters of the simulator.

Real implementations of OppNets are limited to communicating devices handed out by one organizer. One example is the Sámi Network Connectivity project [6] which provides network connectivity to nomadic reindeer herders. The authors of [7] give another example which describes conference badges which are used by researchers to connect and exchange research interests. If there is a match, the user is made aware of the communication partner and a conversation is initialized. Additional information about their research can be exchanged automatically and used by the device owner after the conference. These current use cases only support a limited number of participants.

We use non-rooted off-the-shelf Android devices to establish OppNets on mobile devices and thus opening the opportunity for OppNets with millions or billions of users. Android devices "continue to capture roughly 85% of the worldwide smartphone volume"[1]. *opptain* [3] is an Android-based application to establish OppNets on smartphones and other smart devices. We are working on *opptain* with regard to connection possibilities, routing schemes, forwarding strategies, drop policies, security, and multiple signal way transmissions. We implemented several routing schemes, forward and drop policies and have multiple third-party applications available. Implemented routing schemes are PRoPHET, Epidemic, and (Binary) Spray & Wait/Focus [8,9]. The third-party applications can use the provided API to use the opptain network; under ongoing development are catastrophe, chat, file sharing, and gaming applications, as well as a distributed database. This shows the wide range of applications possible to run through this local, trace-less communication.

To test opptain and to create automatically running field tests, we developed a test framework application. This framework helps to distribute a settings file to all test devices opportunistically. All devices start the test at the same time, and after the test period, all individual results are aggregated on one device for evaluation.

The goal of this paper is to show that Android-based OppNets are capable of successfully transmitting and delivering data such as chat messages or files. By establishing such an OppNet, messages and files that are not time crucial can be

[1] See https://www.gartner.com/newsroom/id/3859963.

transmitted locally and without an Internet connection. This is interesting for a variety of situations.

During an *office scenario*, for example, there are both messages and files that are not time crucial among others that cannot be delayed and have to be transmitted in almost real-time like in prioritized emails. The former mentioned messages and files, however, can be transmitted via our OppNet.

Another example is a *catastrophe scenario* where there may be a loss of Internet connection to communicate with each other. Crucial in this scenario would be people in need of help which could be asked for with the help of an OppNet built up by smartphones from exactly those people involved. A message or rather request for help could be either answered or carried along by every participant in range and later on received by another person and delivered successfully respectively.

Also, our OppNet is applicable for situations with no infrastructure in general like areas with strict Internet surveillance going on, where communication must be hidden or is otherwise blocked. In such a *censorship-risky scenario*, people can communicate through an OppNet to stay connected and self-sufficient.

opptain may also be deployed in great rural areas without infrastructure. In a *village scenario*, there is a large area to cover for transmitting messages between villages or small towns. In this use case, delay is not crucial considering inter-village movement is rather slow.

The contributions of this paper are the following:

– We developed a methodology for evaluating Android-based OppNets.
– We developed a test framework for the evaluation, for which different routing protocols, forward and drop policies, TTLs or many different OppNet-related variations can be tested.
– We ran an initial field test with 26 devices. We created a testbed of devices and ran it with real people in our university building representing a use case.
– We present and discuss the findings of this field experiment.

The results show that we can use an Android-based OppNet to forward information in office scenarios. We discuss how we can use these results for the prediction of other scenarios, like catastrophe situations.

2 Methodology for Evaluating Opportunistic Networks

In this section, we present the methodology to evaluate OppNets in general, and specific for Android-based OppNets. We define the metrics that are used to evaluate OppNets and the message states to determine those. After that, we define our experimental setup to test the opptain application in a field experiment.

At the end of this section, we defined our test setup and metrics to present the evaluation in the next section.

2.1 Message States

In this section, we define the state of a message in the network. In OppNets a message can either be at the sending node, a relaying node, or the destination.

The essential *message states* are the following:

1. *'generated'* describes that a message is generated at some node.
2. *'received'* describes that the message is forwarded to a relay node but not the destination.
3. *'delivered'* describes that the message reached its destination.
4. *'reacted'* describes that a new message is generated in reaction to a delivered message. The delivered message's origin serves as the new recipient.

Both, simulators and our test framework track these message states as input for the metrics' formulas. We show these metrics in the following section.

2.2 Metrics

In this section we present the metrics for evaluation of our network. We identify *Delivery Ratio* and *Delay* as the primary metrics and *Overhead Ratio* and *Hop Count* as secondary metrics. While these four metrics can be used in OppNets generally, two additional metrics, *Client Time* and *Hotspot Time*, are important metrics for our specific Android-based OppNet.

The metrics used in this paper are the following:

1. *Delivery Ratio* is the ratio between delivered messages and generated ones. Therefore, it is calculated with
$$DR = M_{Delivered}/M_{Generated} \tag{1}$$
where M is the sum of messages and DR is the calculated delivery ratio. As there are no end-to-end paths in OppNets, message delivery is not guaranteed. Therefore, *Delivery Ratio* is an essential measure for OppNets.
2. *Delay* is the time that successfully delivered messages need to reach their destination. Therefore, it is calculated with
$$D = TS_{Delivered} - TS_{Generated} \tag{2}$$
where TS is the time in seconds and D is the calculated delay. As messages are relayed in store-carry-forward fashion *Delay* is used to measure the quality of the network.
3. *Overhead Ratio* is the ratio between transmissions and delivered messages. Therefore, it is calculated with
$$OR = M_{Transmitted}/M_{Delivered} \tag{3}$$
$$M_{Transmitted} = M_{Received} + M_{Delivered} \tag{4}$$
where M is the sum of messages and OR is the calculated overhead ratio. We track *Overhead Ratio* as most routing protocols create message copies and all copies are transmitted more often than necessary.

4. *Hop Count* is the number of hops that are necessary to deliver a message.
5. *Client/Hotspot Time* is the duration a device is either hotspot or client. Android devices are not able to connect in ad-hoc fashion but use the Wi-Fi infrastructure mode [3]. Due to these limitations a device is either a tethering hotspot or acts as a client, and the devices have to be in different states to transfer data. The duration of *Client Time* and *Hotspot Time* show how long the devices are in each state, respectively.

2.3 Setup of Field Test

In this section, we describe the setup of our field tests. The goal of this evaluation and field test is to show how an Android-based OppNet performs in a typical office scenario with reasonable demand for data exchange. An office scenario provides us the possibility to test our network application in a realistic environment. We handed out 26 pre-configured, unrooted Android mobile devices, out of which 18 were mobile and eight static during office time.

With the help of our test framework, it is possible to set parameters in opptain. Possible settings are routing protocol, TTL, message size, the time interval in which messages are generated, the probability to *react* to a delivered message. These settings can be changed in a simulator as well so that a supportive comparison in a simulator is doable. A simulator could also emulate the network and opportunistic meetings, based on our field study. In the field test, the devices are spread to users. Our test framework can aggregate the data and log the message states on sending, relaying and receiving devices. Thus, the field tests could be reproduced in simulation.

In our test setup, we choose an overall test duration of five hours since these are the core hours of an average office day in our test environment. Crucial during this time are TTL and routing protocols as those are varied parameters. For routing, we choose either Epidemic [10] or PRoPHET [5] with a *routingMinP* of 0.4 and 150 and 300 min TTL respectively. The randomly generated messages are at a fixed size of 10 Kb. The response probability is set to 70 which implies that there is a 70% chance that there will be a direct reaction to a delivered message.

We choose to model the network into four communication islands (subnetworks) to simulate opportunistic behavior through individual offices. Islands 2 and 4 each consist of four devices whereas island 1 consists of ten and island 3 of eight devices. Communication between islands is only possible if the participants move around within the range of other participants' devices.

We chose to run two field tests simultaneously. Both networks are defined that only devices of the own network can connect to each other. Running two field test at the same time has two advantages: First, we can cope with possible outages of devices. Second, we can compare two test runs with the same parameters, pattern of movement, and social interaction of the participants. A disadvantage might be the WiFi interference of the networks to each other, but in a typical office scenario, there already is WiFi interference.

Table 1. Numerical results

Network	Min	Max	Avg
All	0.48	15844.26	1375.03
Reacted	0.48	15519.59	660.77
1	0.57	15519.59	842.99
2	0.90	2327.90	109.55
3	1.34	8351.99	621.41
4	2.60	5743.31	203.13

(a) Delay (s)

Network	Min	Max	Avg
All	38.31	71.44	60.85
Reacted	67.90	86.66	79.60
1	49.15	86.61	75.25
2	0.00	100.00	68.88
3	6.52	80.79	51.22
4	0.00	100.00	85.94

(b) Delivery Ratio (%)

Network	Min	Max	Avg
All	4.44	6.39	5.15
Reacted	2.58	3.78	3.11
1	4.05	9.34	6.32
2	2.48	10.20	4.83
3	2.56	30.00	8.63
4	2.95	10.91	4.99

(c) Overhead Ratio

Network	Min	Max	Avg
All	1.00	7.00	1.97
1	1.00	4.00	1.39
2	1.00	1.00	1.00
3	1.00	3.00	1.24
4	1.00	1.00	1.00

(d) Hop Count

Network	Min	Max	Avg
All	0.099	328.028	43.472
1	0.099	325.007	48.453
2	9.803	40.936	22.923
3	0.109	328.028	49.790
4	15.271	57.210	34.745

(e) Client Time (s)

Network	Min	Max	Avg
All	30.014	234.191	53.476
1	30.069	234.191	57.500
2	30.014	77.045	47.219
3	30.017	137.158	53.655
4	33.013	70.048	48.026

(f) Hotspot Time (s)

3 Performance of Real World Opportunistic Networks

In this section, we evaluate the performance of real-world OppNets based on the metrics described in the last section. In the end, results are related to one another and discussed.

The measurements indicate as the main result that the OppNet for our test purpose is at all events capable of reliably transmitting data with an average delay of 1375.03 s and a delivery ratio of 60.85%. In the following, we depict the results of our field tests. Note that we evaluate the overall average of all field tests merged with all sub-networks and additionally we evaluate the overall average of all field tests partitioned by every single sub-network.

Figure 3 shows results separated by test runs. The first two characters indicate the number of the test and its duplication; the three digits indicate the TTL of 150 and 300 min respectively. The last character indicates the routing protocol used for testing (either [e]pidemic or [p]RoPHET).

Thus, we can analyze every single sub-network completely encapsulated from each other without intersection (see Table 1). In this case, we consider only messages *generated*, *received*, *delivered* and *reacted* inside this sub-network. Therefore, these messages are just a subset of all messages and no inter-sub-network transmissions are considered. That is why it is not possible to compare the overall overhead ratio of all field tests to that of only a single sub-network.

Delay. On average a message's delay was 1375.03 s (~23 min) from its creation time until it was successfully delivered. The value of the upper bound is 15844.23 s (~4.4 h) and therefore about 11.5 times higher than the average value. With a value of roughly just half of a second, the lower bound shows a rather

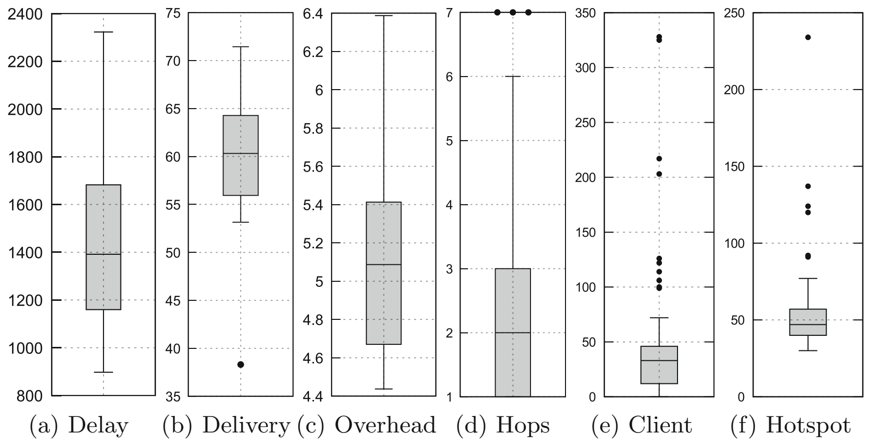


Fig. 1. Box plots of the metrics

fast transmission. Thus, the highest delay of all field tests is almost as long as the overall test duration.

50% of delay per message is in the range of around 1150 s (~19 min) to 1700 s (~28 min). The upper 25% of delay is between approximately 1700 and 2250 s (~37.5 min) (see Fig. 1a).

Sub-networks 1 and 3 show an average delay of 842.99 (~14 min) and 621.41 s (~10.3 min) respectively whereas sub-networks 2 and 4 present a faster transmission (see Table 1) of 109.41 (~1.8 min) and 203.13 s (~3.4 min). Also, the upper bounds correspond in this manner. As of sub-networks 1 and 3, there is a maximum delay of 15519.59 and 8341.99 s. Sub-networks 2 and 4 show us a delay of 2327.90 and 5741.31 s.

Delivery Ratio. Having a closer look at the delivery ratio we see that of all generated messages on average 60.85% arrived at their randomly selected destination. The upper and lower bounds are at 71.44 and 38.31%. 50% of the delivery ratios are between about 56 and 64%, and the upper 25% are in the range of around 64 and 72%. There is one outlier set at the absolute minimum of 38.31% (see Fig. 1b). As for the sub-networks values of the lower bound strongly differ from each other ranging from 0 to 49.15% delivery rate. The upper bound ranges from 71.44 to 100%.

Overhead Ratio. The overhead ratio of all field tests shows us that for every message delivered to its destination 5.15 messages were received by relay nodes on an average. Thus, it appears that for one message to be successfully delivered almost 40% of all nodes received this message. We see that the upper and lower limits are 6.39 and 4.44 respectively. There is no significantly high or low value relative to the average value, so it seems all networks were flooded with messages almost evenly. This appears to be plausible considering that the test framework randomly generates messages in a uniformly distributed span of time. So the more messages are generated, the more can and will be transmitted. Also, since we only use routing protocols on the basis of flooding messages are spread very broadly. This behavior was presumed during preparation. 50% of all overhead is set between 4.8 and 5.4 generated messages per delivery. 25% set from 5.4 to 6.39 messages (see Fig. 1c).

As for the sub-networks, there are more broad ranges from lower to upper bounds. At a maximum, there is an overhead of 30 messages (4.67 times the overall average value). At a minimum, there is an overhead of 2.48 messages (1.73 times the overall average value). The average values differ by 3.48 messages (sub-network 3) at maximum and by 0.16 (sub-network 4) at minimum. It is evident that sub-network 3's upper limit is quite elevated compared to the corresponding average's value (see Table 1) That is because numerous messages did not reach their destination. However, this is only the worst-case of all field tests and the average value distributed across all field tests is much more significant. Such a peak might happen inside an OppNet since the results of our study are entirely dependent on human movement and interaction with each other.

Hop Count. The average hop count is oscillating around two hops and is almost constant through all field test. The maximum value over all four sub-networks ranges from 1 to 7. Half of all hops of all networks and all tests are in the range of 1 to 3 (see Fig. 1d). While 22.5% of all hops are in the range of 3 to 6 hops, only 2.5% of all hops are placing at 7 hops. Sub-networks 2 and 4 have an average of 1 hop, which is the only possible option since each sub-network consists of two devices.

Client Time Versus Hotspot Time. The overall time a device resides in client state is about 43 s compared to about 53 s in hotspot state (see Fig. 1e and f). In sub-networks 1 and 3 there is a maximum of 325 and 328 s (~5 min) respectively in client state and for both sub-networks a minimum of one-tenth of a second. This seems to be a very small value and is due to the mode of operation of opptain which may not be able to establish this state and continue to its next one entirely. In hotspot state, there is a maximum of 234 and 137 s and a minimum of 30 s.

50% of all devices reside in a range of about 10 to 49 s in client task. The upper 22.5% range from 49 to about 75 s whereas the top 2.5% of all client times range from 100 to a maximum of 328 s.

50% of hotspot times reside in the range of roughly 48 to 53 s. Thus, this task is oscillating around 50 s at an average. The upper 22.5% range from about 53 to 80 s. However the last 2.5% of the upper bound ranges from about 95 to 234 s.

In sub-networks 2 and 4, the ranges of client task and hotspot task combined set from 22.923 s to 48.026 s. Thus, sub-networks 2 and 4 show a more narrow span of time in contrast to sub-networks 1 and 3.

Table 2. Numerical results of message states

Nw.	Min	Max	Avg
All	539	726	663.38
1	63	102	84.13
2	4	9	5.75
3	41	60	50.88
4	5	15	8.63

(a) Generated Messages

Nw.	Min	Max	Avg
All	2344	4778	3637.62
1	602	1314	1068.5
2	0	129	75
3	121	779	395.63
4	2	124	102.5

(b) Received Messages

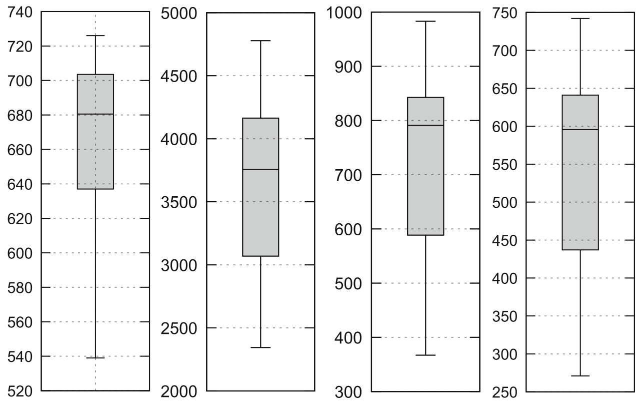
Nw.	Min	Max	Avg
All	367	983	724,25
1	298	488	399,63
2	0	128	74,38
3	80	382	223,75
4	2	123	102

(c) Delivered Messages

Nw.	Min	Max	Avg
All	271	742	545
1	43	234	137,88
2	0	38	15,25
3	5	119	57,13
4	0	30	16,88

(d) Reacted Messages

Message States. Figure 2 and Table 2 show a distribution of the four predominant states a message can be in. As for generated messages, there is an average of 663.38 over 5 h test duration. Thus, about 2.1 messages per minute are generated. On an average 545 messages are resent as a reaction of delivered ones. Roughly 720 messages are delivered. Included in those are also messages reacted on. With an average of 3637.62 messages, received messages are about 5.5 times the size of generated ones.



(a) Generated (b) Received (c) Delivered (d) Reacted

Fig. 2. Number of messages for each state

Reacted Messages. Delivered messages are reacted on at an average of 75.25%. That produced an overhead of 3.11 received messages per delivery. Thus, that is nearly two-thirds of the overall overhead ratio (see Table 1). Messages reacted on are delivered at an average of 79.6% with a delay of 660.77 s (~11 min).

These numbers allow identifying possible use cases of OppNet-based applications on Android-based smartphones in an office environment.

4 Discussion

In this section, we correlate the previously described outcome of our field tests. First, we compare the individual test runs among each other emphasizing commonalities and differences, followed by a general discussion of all results.

A test run with a TTL of 300 min was expected to be much worse concerning overhead ratio than a run with a TTL of 150 min as there would be much more transferable messages inside the network. Both are conducted with the Epidemic routing protocol. The equivalent test runs *1A-150e* & *1B-150e* and *3A-300e* & *3B-300e* show however that this assumption cannot be verified evidently.

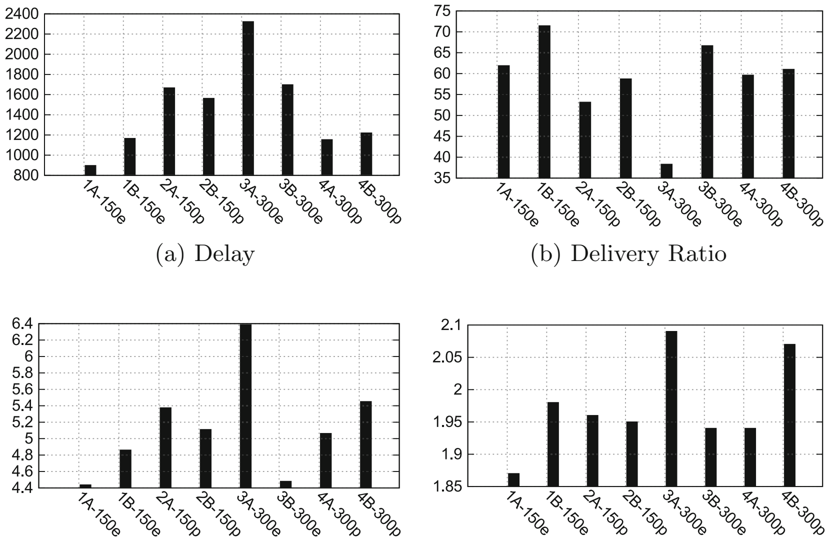
Epidemic routing supposedly produces a higher overhead than its modified version PRoPHET which uses a threshold value and algorithms to determine a suitable node to forward a message to. Also, it was expected that PRoPHET presents a superior ratio between overhead and delivery meaning there would be the same delivery ratio with less overhead ratio. These assumptions are also not supported.

Test run *3A-300e* presents the highest values in overhead ratio, delay and hop count plus the lowest delivery ratio (see Fig. 3). In theory, a high overhead ratio can be linked to an elevated hop count and slow transmission. We would expect both high delivery ratio and high overhead ratio combined since the chance would be higher to deliver a message with an elevated overhead ratio.

The delay of *1A-150e* and *1B-150e* is very low in comparison to the other test runs. This can be explained with the possibility that the TTL may run out and messages eventually are not delivered. The high overhead ratio of *3A-300e* and *3B-300e* is an indication for this assumption. Also, there is a possibility that the client tasks and hotspot tasks did not run in favor of each other. It is an issue of smartphones that two devices may not be able to connect to each other because a client may only connect to a hotspot and not to another client. Nevertheless, this is the only option for unrooted devices to interact with each other without user interaction. It is the same with a hotspot that cannot connect to another hotspot.

It is striking that test run *3B-300e* presents the lowest overhead combined with the highest delivery ratio while supposedly being the same as *3A-300e* in configuration and pattern of movement. This vast difference in delay and overhead can be explained by the fact that one device failed during test run *3B-300e*. Thus, potentially relevant data is lost, and results are tampered. The failed device taken in account, the delivery ratio would be equal or even higher than it already is. It is also important to see how important the message ferry devices are in such networks.

In the test runs *1A-150e* and *1B-150e*, overhead ratio, delivery ratio and delay are all roughly similar to those of test runs *4A-300p* and *4B-300p* respectively. This is quite interesting since the first tests are performed with a TTL of 150 min and with Epidemic routing as the last two tests are set up with 300 min of TTL and the PRoPHET routing protocol. Thus, both groups of tests run on an entirely different configuration.



(a) Delay (b) Delivery Ratio

(c) Overhead Ratio (d) Hop Count

Fig. 3. Evaluation analysis according to the various setups

Almost the same results are presented looking at Table 1. Reacted messages are delivered at a maximum rate of almost 90% plus with nearly half of the overhead of all messages and with a comparatively low delay of 400 to 600 s.

These results are a strong indication that both setups fit quite well for a use case in a scenario in which messages are exchanged continuously and are supposed to be reacted on, such as in an office scenario.

5 Conclusion

This section concludes the previously presented results and correlates these with realistic scenarios and use cases of OppNets.

Our evaluation shows that our smartphone-based OppNets can successfully deliver data such as chat messages or files. We show that it is possible for participants to engage in a conversation and keep it maintained, though, with a certain delay. Thus, OppNets can practically be used to sustain a local communication structure with specific use cases. The results pose possibilities for a few real-life scenarios like an average office day, catastrophe or censorship situations or the village scenario. The used parameters may be applied to all of those situations.

Since it is essential that the vast majority of messages are delivered, we propose to neglect overhead ratio in favor of delivery ratio. As we can see from the results, all proposed scenarios rely on the social interaction of people in general. Therefore, our OppNet only works through people's movement and performs even better with an equal distribution of participants inside the network area.

Our OppNet may not be suitable for real-time transmission (for example VoIP) since it is not guaranteed that there is a stable and constant connection at all times. However, overall delay in a chat or mail conversation generally poses no significant problem as long as time is no crucial factor.

For the future, we plan further testing in larger scenarios with the applications sketched in the introduction. This includes varying the message size, the message creation frequency, and the mobility patterns to support more scenarios. There are many parameters in general which can be adjusted and have to be evaluated.

References

1. Cheraghi, A., Amft, T., Sati, S., Hagemeister, P., Graffi, K.: The state of simulation tools for P2P networks on mobile ad-hoc and opportunistic networks. In: IEEE ICCCN 2016: Proceedings of the International Conference on Computer Communication and Networks, pp. 1–7 (2016)
2. Graffi, K.: PeerfactSim.KOM: a P2P system simulator - experiences and lessons learned. In: IEEE P2P 2011: Proceedings of the International Conference on Peer-to-Peer Computing, pp. 154–155 (2011)
3. Ippisch, A., Graffi, K.: Infrastructure mode based opportunistic networks on android devices. In: 2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA), pp. 454–461 (2017). https://doi.org/10.1109/AINA.2017.32
4. Keränen, A., Ott, J., Kärkkäinen, T.: The ONE simulator for DTN protocol evaluation. In: SIMUTools 2009: Proceedings of the 2nd International Conference on Simulation Tools and Techniques. ICST, New York, NY, USA (2009)
5. Lindgren, A., Doria, A., Davies, E., Grasic, S.: RFC 6693: probabilistic routing protocol for intermittently connected networks. IETF (2012)
6. Lindgren, A., Doria, A., Lindblom, J., Ek, M.: Networking in the land of northern lights: two years of experiences from DTN system deployments. In: Wireless Networks and Systems for Developing Regions (2008)
7. Mühlhäuser, M.: Handbook of Research on Ubiquitous Computing Technology for Real Time Enterprises. Handbook of Research On... Series. IGI Global (2008). https://books.google.de/books?id=PQjsqETjwhYC
8. Spyropoulos, T., Psounis, K.: Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In: Proceedings of the ACM SIGCOMM Workshop on Delay-Tolerant Networking (2005)
9. Spyropoulos, T., Psounis, K., Raghavendra, C.S.: Spray and focus: efficient mobility-assisted routing for heterogeneous and correlated mobility. In: Proceedings of the IEEE International Conference on Pervasive Computing and Communications - Workshops (PerCom Workshops), pp. 79–85 (2007)
10. Vahdat, A., Becker, D.: Epidemic routing for partially-connected ad hoc networks. Technique report, Department of Computer Science, Duke University, USA, vol. 20, no. 6 (2000)