Smartphone-based Frameworks and Protocols for Opportunistic Networking

Inaugural-Dissertation

zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Andre Ippisch

geboren in

Düsseldorf

Düsseldorf, November 2018

aus dem Institut für Informatik
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Berichterstatter:

1. Jun.-Prof. Dr.-Ing. Kalman Graffi

2. Prof. Dr. Michael Schöttner

Tag der mündlichen Prüfung: 16. Januar 2019

Für meinen Fuchs...

Abstract

Opportunistic Networks are mobile, delay-tolerant networks with intermittent node contacts in which messages are transferred with the *Store-Carry-Forward* principle. In these decentralized networks, which stand in contrast to the Internet's infrastructure, devices are connected to each other instead of an access point. Many of the current smartphones and apps heavily rely on and are rather useless without Internet access, but this access is not always guaranteed and can be unavailable due to catastrophes, censorship or just dead spots. Additionally, the usage of the Internet might be unwanted because of its dependency on hardware and lack of privacy. Hence for smartphones, Opportunistic Networks can offer a useful addition or alternative to the Internet. Because there are parallels between the movement of people and the dynamics in Opportunistic Networks, and additionally, most people own a smartphone, they have the potential to be a suitable platform for Opportunistic Networks. However, there has not been much work on this combination, and in fact not much work on the use of Opportunistic Networks themselves in practice. To change that, several challenges have to be resolved to establish an Opportunistic Network with smartphones and provide the network application *opptain*.

Building an Opportunistic Network for smartphones requires solutions for the basic challenges of node discovery, connection establishment, and one- and multi-hop communication. To allow a wide applicability of our networking mechanisms, we must use off-the-shelf and unrooted smartphones, possibly with the most widespread operating system. Our first goal is to discover other participating smartphones and to connect to them automatically, which allows our application to create the network without user interaction. Because current smartphones lack wireless ad-hoc network technologies, we present an approach called *Hotspotting* in which the device scans for tethering hotspots of other network participants in the surroundings and, after a successful search, connects to one, else the device itself becomes a hotspot and lets other devices connect to it.

Once the network and one-hop communication is established, our next goal is to enable multi-hop communication with routing protocols that manage the dissemination of the messages in the network. We include routing protocols proposed in literature in our network and improve on them with new replication control strategies. During the establishment of a connection, meta data that is needed for these protocols is gathered and exchanged between two communicating nodes. After the two nodes connect, each node has to decide which messages to forward or replicate to the connected node. Either the nodes always replicate all messages, i.e. flood the network, or local information and the aforementioned meta data are used for repli-

cation decisions of more sophisticated routing protocols. To improve the one- and multi-hop communication, we enhance the meta data exchange and use smartwatches as a second signal way.

When nodes can connect and messages are routed through the network, there are still open challenges. To avoid overwriting any user data in the smartphone's shared storage with network messages, we develop a specific buffer management. To allow for the correct processing of the network's time specifications that are exchanged between devices, for example with the remaining *Time-To-Live* of our messages or the global order of messages, we also develop a specific time management procedure.

Having put together all the basic requirements for a working Opportunistic Network, we consider other dynamics that affect the stability of the network, in particular, node density and security. Node density has a high impact on the network since messages are transferred from one device to another, and therefore devices that run our application must meet regularly to establish a functioning network. To show the impact of security in decentralized networks, we conduct a survey of the possible threats and attacks that affect single and multiple devices, in which we also show the proposed solutions and mitigation techniques. Lastly, to evaluate our work, we conduct a field experiment which provides promising results that show that our application can establish an efficient Opportunistic Network.

Acknowledgments

Mit großer Freude nehme ich mir hier den Platz, um mich bei den Menschen zu bedanken, die mich während meiner Promotionszeit begleitet und unterstützt haben, und um ein paar Dinge während dieser Zeit zu würdigen.

Ganz besonders bedanken möchte ich mich bei meinem Doktorvater Kalman für die Möglichkeit und Herausforderung dieser Promotion, die Unterstützung und die Freiheiten. Außerdem danke ich meinem Mentor Michael für unsere Gespräche über Forschung und Lehre sowie für die Inspiration für meine Android-Programmierung-Vorlesung.

Ich möchte mich bei meinen Vögeln – Daniel, Raphael und Tobias – bedanken; für die zig Jahre Spaß, die wir hatten und hoffentlich noch sehr lange haben werden. Bedanken möchte ich mich bei allen Kollegen vom Rechnernetze-Lehrstuhl und der Technik sozialer Netzwerke-Arbeitsgruppe für die wunderbare Zeit und die tolle Atmosphäre. Vermissen werde ich #1120, die Wutausbrüche von Wegi, die Begeisterungsstürme von Christian, das Füßeln mit Amfti, Distis Geier zu sein, das “ok ok ok” vom einen Ahmad und das klingelnde Handy vom anderen Ahmad, die *thumbs up* von Newton, der denglische Raed und natürlich Salem und seine doppelten Leerzeichen. Ich bedanke mich bei unseren kompetenten und hilfsbereiten Sekretärinnen – Sabine, Angela, Claudia und Claudia – und Systemadministratoren – Thomas und Guido –, die mir immer geholfen haben und mit denen ich mich immer super unterhalten konnte.

Was diese Zeit so vielfältig gemacht hat, waren auch die ganze Dinge drumherum, wofür ich ziemlich dankbar bin und die ich gerne würdigen möchte. Die Kickerrunden mit dem STUPS-Lehrstuhl, die Fachschaft Informatik, die nicht immer nur Lärm gemacht hat, sondern auch mal gerne geholfen hat, die Fachschaft Physik und die InPhiMa, die für das nötige Zusatzprogramm gesorgt haben, die Schweinemensakapelle, die sogar Absolventenfeiern rocken durfte, und die *Straße* und den *Hass*; keine Angst, lieber Leser, wir haben keinen geheimen Fight-Club gegründet.

Ich bedanke mich auch bei allen Studierenden, die mich unterstützt haben, sei es durch Bachelor-, Projekt- oder Masterarbeiten oder durch eine Hiwi-Tätigkeit. Ein großer Dank geht dabei an Jannik, Tobias, Philipp, Bashkim, und Martin, die auch über diese Arbeiten hinaus weitergemacht haben, um unsere Forschung voranzutreiben. An dieser Stelle möchte ich auch ausdrücken, dass es mich sehr freut, zu sehen, wie viel Energie Studierende in Projekte stecken, wenn sie ihre eigenen Ideen umsetzen dürfen und Spaß an dem haben, was sie machen.

Zum Schluss möchte ich mich bei meinen Eltern und meiner Freundin Jessy bedanken; ohne viele Worte, denn diese könnten nur einen Bruchteil davon ausdrücken, wie dankbar ich für die Unterstützung bin.

Im Sinne dieser Dissertation hoffe ich, dass wir alle gemeinsam ein großes glückliches opportunistisches Netz bleiben; vielleicht nicht immer und überall direkt miteinander verbunden, aber immer und überall miteinander vernetzt.

Contents

1 Introduction **1**

1.1 Motivation and Problem Statement 2
1.2 Research Questions 6
1.3 Contributions 12
1.4 Thesis Organization 15

2 Android-based Opportunistic Networks **17**

2.1 Infrastructure Mode Based Opportunistic Networks on Android Devices 21
2.2 Time and Space in Android-based Opportunistic Networks 24

3 Studies and Evaluation for our Android-based Opportunistic Network **29**

3.1 The Impact of Node Density and Transmission Range on Opportunistic Networks 31
3.2 Mitigation Techniques for Software- and Network-based Threats 35
3.3 Field Experiment on the Performance of an Android-based Opportunistic Network 39

4 Improving One-Hop and Two-Hop Communication Options **43**

4.1 Contact Matching and Connection Scheduling 45
4.2 An Android Wear OS Framework for Sensor Data and Network Interfaces . . . 49

5 Optimal Message Replication Control **53**

5.1 Dynamic Replication Control Strategy 57
5.2 Replication Probability-based Routing Scheme 61
5.3 Optimal Replication Based on Optimal Path Hops 64

6 Conclusion & Future Work **67**

6.1 Conclusion 68
6.2 Future Work 73

Chapter 1

Introduction

Nowadays, almost every device with a computer chip can be connected to other similar devices and consequently, its users are inherently connected to one another. Much of the communication with friends, family, and colleagues happens online as most of us humans are, often constantly, connected to the Internet through various devices. One invention that makes this permanent networking possible is the smartphone. With a smartphone, we carry a small yet mighty computer around with us. Up-to-date smartphones have powerful processors and graphics cards, large storage, battery, and an abundance of sensors and a *Global Positioning System (GPS)* receiver. They use their many networking possibilities either to access the Internet (*General Packet Radio Service (GPRS)*, *Universal Mobile Telecommunications System (UMTS)*, *Long Term Evolution (LTE)*, *Wi-Fi*), to connect to additional devices like smartwatches and headsets (*Bluetooth*), or for contactless payments (*Near Field Communication (NFC)*). However, apart from fast Internet access due to *LTE* and *Wi-Fi* support, what characteristics of a smartphone really make up the term smart? Vaguely speaking, it is used to separate the newer phones from old ones, with which users were only able to phone and to text; with smartphones, *apps* are key. In most situations where one needs a particular functionality or a possibility to perform a particular task, there is an app for that. But, even if the apps are what distinguishes smartphones from older phones, the features of the apps often heavily rely on the Internet.

The Internet is a centralized network in which services are provided by servers and used by clients. Access to the Internet, either by cable, cell tower or access point, however, is not always guaranteed. Infrastructure might be absent, for example, it might have collapsed due to a catastrophic situation, or it can simply be nonexistent. In these situations, current smartphones or rather current apps become almost useless.

Opportunistic Networks [1] stand in contrast to the Internet. Devices in these networks are connected to each other instead of to an access point, the network is completely decentralized, and there is no guarantee of an end-to-end connection. Routing in these networks is realized through the *Store-Carry-Forward* mechanism: The nodes are so-called data ferries that store messages in their buffer and carry them around until the messages either reach their destination, are forwarded to another node on the way to the destination or have to be dropped due to full storage. Until now, *Opportunistic Networks* have mostly been examined in theory or in simulators. But, there are parallels between the movement of people and the dynamics in *Opportunistic Networks*, since most of us follow social patterns that allow us to be in groups, leave those groups to move somewhere else, and again form new groups, which is what a node in an *Opportunistic Network* has to do to disseminate messages. We thereby are the perfect carriers for nodes of an *Opportunistic Network*, yet this far, the technology was lacking to allow us realize the full advantage of such a potential.

In this dissertation, so as to provide current apps with an alternative to the Internet, we want to combine the technical capabilities of our smartphones, the social behavior of us human beings, and the principles of *Opportunistic Networking*. Next, we motivate our objective and specify which actions we have to take to achieve it.

1.1 Motivation and Problem Statement

In the last years, mobile devices like smartphones, tablets, and smartwatches have become more and more popular and omnipresent. The key motivation for this dissertation is to use the potential of mobile devices and integrate them into a human-controlled real-world *Opportunistic Network*. There are many reasons for the idea of a real-world *Opportunistic Network* for smartphones as an alternative to networking via the Internet. As mentioned before, the first reason is the argument that Internet access is not guaranteed or can be restricted. The Internet might not be available due to catastrophes that destroy the infrastructure or due to censorship as it was the case in the Arab Spring when the Egyptian Government cut most of the country off the Internet. There are also parts of the world, particularly areas considered to lie in the global south, where Internet access is nonexistent to begin with, although villages might want to be connected to each other. Dead spots, limited data volume and closed buildings may restrict the Internet access even in well covered areas.

Apart from the thought that the Internet is not always available, there are also arguments for *Opportunistic Networks* originating from the wish to explicitly not use the Internet. First, there is the privacy argument. With most current messaging apps for example, if you want to send a picture to the person standing right next to you, the communication will nevertheless be handled via a server on the Internet that might not even be located in the country you are in. The messages leave a trace on the Internet that is undesirable and can be avoided with local communication. This local communication can also happen to be much faster than the routing through the Internet, which is expressed by the performance argument. Economically speaking, local communication is in no need of any infrastructure or any hardware apart from the mobile device, hence it is free of charge; one does not need an expensive and limited data plan. The next and last argument is about sustainability. Less required hardware means less fabrication and energy usage, which is beneficial for the preservation of the environment.

While all these arguments motivate the adoption of *Opportunistic Networks* for smartphones, we also want to state the reasons for using smartphones for these kind of networks. Imagine an *Opportunistic Network* that works with a specific hardware that is only designed for this network. How many people are willing to buy this specific piece of hardware to use a network technology that they have never heard of? But, almost everybody owns a smartphone and distributing new software via the app store is quite simple, thus getting people to use an *Opportunistic Network* is way easier when offering it on their smartphone. Additionally, the observed mobility patterns of people in the formation of groups and their consequent movements to other groups suits an *Opportunistic Network* very well.

For all these listed arguments it is reasonable to build *Opportunistic Networks* using smartphones; however, there is a lack of apps supporting this idea. To put the idea into practice, we have to face multiple challenges. At first, we have to decide on a mobile operating system which enables us to use the smartphone as a node in our network. Subsequently, we must develop communication protocols and policies to assure that the *Opportunistic Network* is not only usable but also efficient. Once this is accomplished, we need an application that actually establishes the network. Additionally, our goal is to create such an application not only for real-life usage but also as a playground for us and other researchers and developers as well. Literature proposes a variety of protocols for *Opportunistic Networks* that have been solely evaluated in simulators and now can be analyzed and improved with our application as the test environment. By this means, our application benefits not only the work on the combination of smartphones and *Opportunistic Networks* but also the research on *Opportunistic Networks* per se.

To establish such an *Opportunistic Network* with smartphones, we take a look at the following problems:

Choice of Operating System & Connection Technologies First, we have to find a smartphone operating system which can connect smartphones automatically and without user interaction. When we look at the smartphone market, while there is a multitude of manufacturers who produce billions of smartphones every year [2], there are only a handful of operating systems for these smartphones, because many manufacturers use Google's Android. Other more or less widespread operating systems are iOS by Apple, Windows Phone by Microsoft, and Blackberry's operating system. All these hardware devices and operating systems have advantages and disadvantages in regard to market share, development and distribution of applications and, most important for our cause, the connection technologies. With these possibilities in mind, we should aim to choose a combination of hardware device, operating system, and connection technology which offers suitable mechanisms for node discovery and connection establishment. When the smartphones can discover other nodes, connect to them, and exchange messages in a one-hop neighborhood, the next problems concern the multi-hop routing of messages.

Replication of Messages & Meta Data Exchange Messages in *Opportunistic Networks* are routed through the network in *Store-Carry-Forward* fashion. When two nodes are connected, both nodes inspect their buffer and optionally use meta data from the other node to decide which messages to exchange. The decision about which messages to forward and which not is part of the *routing protocol*, more precisely the *forward strategy*. The order in which messages are forwarded is important as well because a sudden disconnection of the nodes could leave some messages behind. Next, we look at the problem of how to address and identify nodes, and the issue of ensuring messages are end-to-end encrypted.

Identification & Encryption The nodes in our network need a unique identification that messages can be addressed to. The Internet, for instance, uses IP addresses whose assignments are managed by a centralized entity. In our network, however, the nodes are not created by a centralized instance but realized by installing the application, so we are in need of another practice for address allocation. Also, when nodes send messages through an *Opportunistic Network*, the path consists of relay nodes that carry and forward the message until it reaches the destination. To ensure that relay nodes cannot read messages meant for other nodes, the messages have to be encrypted. When trying to solve this problem, we have to consider the

decentralized form of the network. Next, we address the problems that concern the buffer of network nodes which is used to store messages that are routed through the network.

Buffer Management The nodes in our *Opportunistic Network* are not hardware devices explicitly created for this one purpose but smartphones that we simply use for the purpose of being a network node. This means that the storage of the devices is not only available for the software which turns the devices into nodes of our network and uses parts of it as the buffer for messages, but also for the other apps on the device. We should aim for a buffer management that not only serves the purposes of our software but also does not disturb the user's everyday workflow by overwriting any data or cluttering up the storage. This is why we have to delete messages from the network-specific part of the storage if either the network buffer or the other apps need space. Another case in which messages are deleted from the buffer is when a message expires due to their *Time-To-Live*. The creation time and *Time-To-Live* of a message have to be stored on multiple devices and can involve issues regarding the time management in the network; these issues are presented next.

Time Management Proper dissemination of messages involves the correct calculation of a message's *Time-To-Live* as well as the right order of messages at each device. For this, clocks must either be synchronized or the nodes at least need a system to track the difference of the clocks on the path of a message. Nodes in any network, however, might have unsynchronized clocks, and in a centralized network, the solutions to the problem of clock synchronization are trivial. However, in a decentralized network like our *Opportunistic Network*, synchronization is a problem, as there is no singular correct clock. The messages which are routed through the network are typically created by user applications, which we address next.

User Applications If there is a smartphone-based *Opportunistic Network* and there are no applications utilizing it, then the network itself will not be used. This is obviously a problem we have to resolve by implementing a few fundamental user applications for messaging or sharing files. Also, we should offer a solution that allows third parties to develop their own apps which utilize the network. Next, we look at the problem how we can distribute the applications for our network.

Distribution of our Applications Smartphones are smart because of their apps, which are distributed through app stores. These app stores are offered by Google and Apple for Android

and iOS devices respectively and are centralized instances operating over the Internet. Our network application enables smartphones to participate as nodes in our decentralized *Opportunistic Network* and should therefore also be, even if only additionally, distributed without a central instance. A central instance, to be specific, might not be available during a catastrophe and it might not be desirable to visit in case of a censorship scenario. Our aim is to give the network application the ability to replicate itself to other devices, with the consent of their users, of course. This should be possible with the user applications as well. Finally, the exchange of messages between nodes and between network app and user apps can be disturbed by malicious nodes or apps; a problem which we formulate next.

Software- and Network-based Threats If we want to use third-party apps together with our network application, we encounter a few security risks. These third-party apps have to run on the same smartphone, which, as always, implicates the risk of other apps spying on our application to gather user data or information about our network. While this is generally a problem for all applications, it is particularly important for us to inhibit espionage, as for instance journalists or activists might want to use our application for censorship reasons. Besides spies, there might be evil nodes trying to attack or harm our network or pretend to be a harmless node to disrupt the system. We have to look into attacks, vulnerabilities, and threats against our *Opportunistic Network* and try to find solutions and mitigation techniques.

Section Conclusion

In this section, we motivated the establishment of an *Opportunistic Network* with smartphones and formulated problems that we have to face. We can use the devices that we mostly carry around with us permanently to build a network that can be used for situations in which a centralized network is not desirable or an infrastructure is not available (anymore). To achieve this, we formulate more precise research questions and challenges for the above-mentioned problems in the next section.

1.2 Research Questions

An essential part of this thesis is the identification and formulation of detailed research questions and challenges that have to be overcome to combine mobile devices like smartphones and

Opportunistic Networks. These research questions and challenges, derived from the problem statements in the previous section, are now presented to facilitate the discussion of the course of action and to support the understanding of the significance of the provided solutions.

The Creation of an Android-based Opportunistic Network

The main challenge of this thesis focuses on the creation of an *Opportunistic Network* on smartphones that are controlled and led by human beings and therefore use the same mobility patterns as human beings.

Before addressing the general challenges of establishing an *Opportunistic Network*, we encounter the following questions arising from the usage of smartphones: What are the demands for the combination of mobile devices and *Opportunistic Networks*? The most rudimentary demand for nodes in *Opportunistic Networks* is the feasibility to establish connections to each other. Furthermore, these connections should happen automatically and not require any user interaction. Smartphones should not be *rooted* so that the devices can be used off-the-shelf. Which devices and which operating system do we choose to meet these demands? While working on this thesis, only two operating systems are popular enough to be considered: Android OS (Google) and iOS (Apple) (other operating systems have a current market share of under 0.1%) [2]. Which operating system, Android or iOS, do we choose to meet our demands? What connection technologies do they offer and are they suitable for us? As more research questions arise from the choice of operation system and connection technology, at the very onset, we dictate the use of Android and the *Infrastructure Mode Wi-Fi*. The reasons for this selection are given in Chapter 2.

After having answered the question regarding the operating system and having decided on using Android smartphones with *Infrastructure Mode Wi-Fi*, how do we integrate the technique to build the network? Do we develop an app or can we adapt the operating system? The adaptation of the operating system would make porting our functionality to another smartphone way too complicated for an average smartphone user. This is why we decide on providing an app. Our goal to offer a solution that allows third parties to develop their own apps which utilize our network leads us to the following basic structure: We provide a network app as well as an *Application Programming Interface (API)* that can be used by other developers to access our network. This raises new research questions: Without relying on the Google Play Store, how can we distribute our network app? Besides, what kind of user applications do we develop and make available?

Next, we have to overcome the challenge of creating an *Opportunistic Network* in general, which takes up the following sub-challenges. First, neighbor discovery and connection establishment, second, the exchange of meta information to select all messages for forwarding or replication, third, the exchange of messages, and fourth, the eviction of messages if the buffer is full [3].

The first sub-challenge lies in the details of the realization of the node discovery and the user-interaction-less connection. To solve this task we have to answer the following questions: How do we discover other nodes? How can the connections be established automatically? The second and third sub-challenges are about the conveyance of messages and meta data and hence consider the *routing protocol*. The following questions arise: How do we route in our network and which *routing protocols* do we choose? Should it be possible to add more protocols later on? Do we provide an interface to make the integration of other *routing protocols* as easy as possible? Furthermore, we have to take the details of the protocols into account. *Routing protocols* consist of the exchange of meta data and a *forward strategy*. Based on the meta data that is gathered from a connected node, this *forward strategy* decides which messages to forward or replicate. If we want to offer the possibility of integrating more *routing protocols*, we have to consider which meta data to store and gather. What meta data is useful, needed or maybe even demanded by certain protocols? For example, if a *routing protocol* requires a node to know its two-hop neighborhood, the nodes have to memorize their previous connection partners. Even if we do not at once include all meta data that might be required at some point, we need a structure for the meta data exchange that is easily extendable. Furthermore, to enable the exchange of messages, they need a clear addressee. Each device must have a unique identifier so that messages can be routed to this device. To enable secure communication between devices, especially when relay nodes forward the message, messages must be secured with end-to-end encryption. Thus, the last question regarding the third sub-challenge is: How do we identify devices and encrypt messages? For the fourth sub-challenge, we need do employ a *drop policy* that answers the question of which messages to delete when the buffer is full.

In Chapter 2, we find solutions for these sub-challenges concerning the automatic and user-interaction-less connection of Android smartphones, the identification of nodes, the encryption and dropping of messages and the routing through the network, and we present a first version of our network application called *opptain*.

Time Management and Buffer Management in our Android-based Opportunistic Network

After the solution for building a framework for *Opportunistic Networks* on Android in the form of an app is presented in Section 2.1, *opptain* still has two major issues: first, the missing clock synchronization and second, the missing buffer management. The clocks of different devices might be unsynchronized, so we have to face problems with time specifications, for example with the remaining *Time-To-Live* of our messages or the global order of messages. In the first version of *opptain*, due to the large storage space on Android devices, the buffer of the device is assumed to be infinite, which is obviously not the case. With the following challenges, we want to tackle these issues to have a fully functioning Android-based *Opportunistic Network* that is not only operable in laboratory settings but also in the field.

Several research questions institute this challenge. Regarding time management, the following questions arise: How can we synchronize the clock in *Opportunistic Networks* in general? Can we synchronize the clock of Android devices? Android can synchronize the clock with the *Network Time Protocol (NTP)* via the Internet and the user can either advise the operating system to synchronize or set the clock himself manually. Since one of our demands is running the application without user interaction, having the clock set manually is not an option. *NTP* is likewise not an option since another of our demands is the independence from the Internet. An idea would be to adapt the clock of the smartphone as the application learns the clock of another device in the network. However, that is not possible, because the developer cannot give an application the privilege of setting the clock. Thus, we cannot employ clock synchronization. So, if we cannot synchronize the clocks directly, how can unsynchronized devices cope with the difference between system clocks to work and handle time specifications properly? Furthermore, when the clock is set manually by the user, Android does notify all apps about the time change but not about the offset. The last question about time management is therefore: How can we figure out the offset after a device's clock has changed? Concluding, this challenge will be to identify time changes in the system and to handle unsynchronized devices in the network.

Regarding the buffer management on Android devices, we have to address the following issues. In *Opportunistic Networks*, *Store-Carry-Forward* is used, and messages are stored in the buffer. A buffer can be empty, full or anything between, but if the buffer is full, one or more messages have to be removed to make space for a new message, either when a message is generated at the own device or when a message is forwarded to the device. But how does the buffer work on Android devices? If our smartphone acts as a node in the network, *opptain* shares the storage with other applications. This Android storage must never be full or near full because if it is,

applications might not work properly. Which storage options on Android should be used for the different stages a message can be in, e.g., at the source node, at a relay node, or at the destination node? When do we remove messages from the buffer? How can we prevent the shared buffer of the Android applications to fill up completely? Concluding, this challenge will be to develop a buffer management that resolves these issues and questions.

With the solutions to the previous challenges, we have a second version of *opptain*, which has all the necessary features to function as an *Opportunistic Network* on Android devices. The nodes can detect each other and connect to each other automatically and without user interaction, messages can be created, stored, forwarded and received without disrupting the device, time changes on the devices are detected and considered, and as a bonus, messages are end-to-end encrypted.

Studies and Evaluation for our Android-based Opportunistic Network

After solving the important challenges of node discovery, connection establishment, identification, encryption, routing, and time and buffer management for our *opptain* application, we want to evaluate this system in an open field test. But before we start the open field test, we want to make sure that our application is secure with regard to software- and network-based threats and vulnerabilities and that we know how many devices we need for a meaningful test.

While traditional networks are connected, i.e. there is a path between source and destination, in *Opportunistic Networks* there might be no end-to-end paths but intermittent connections. Messages are forwarded with the *Store-Carry-Forward* principle and we suppose that a high node density is important for the networks to work as there actually must be nodes in the surroundings that messages can be transferred to. Until now, *Opportunistic Networks* have mostly been tested and evaluated in simulators, where most default scenarios just assume a high node density. However, for real-life *Opportunistic Networks* a high node density is not guaranteed and we need to analyze how a lower density impairs the performance. Due to their wide spread use, a high density of smartphones is given, at least for urban areas, however there is a difference between all smartphones participating in the *opptain* network or just a few. Only if there are enough nodes incorporated in the network inside the test area can messages be carried from source to destination. Thus, before starting the field experiment, we want to evaluate the impact of node density on *Opportunistic Networks*. So, we ask the following research questions.

How does the node density influence the performance of *Opportunistic Networks*? How many nodes do we need for our subsequent field experiment to be meaningful?

So far, regarding security, we have implemented *opptain* in all conscience but only explicitly included end-to-end encryption. To offer participants of our field test the assurance that we let them use a secure application, we want to make sure that this is the case. How do we uncover the possible software- and network-based threats and vulnerabilities? What measures must we take to prevent the attacks and patch the vulnerabilities? So, we should conduct a study about software- and network-based security threats and vulnerabilities and find solutions and mitigation techniques to apply to *opptain* before deploying our application for a field test.

With a secure application and knowledge about the impact of node density in *Opportunistic Networks*, we now can conduct a field experiment to test *opptain*. *Opportunistic Networking* applications are interested in the three quality measures *delivery ratio*, *overhead*, and *end-to-end delay*. *Delivery ratio* is the number of delivered messages compared to the number of generated messages, and in an optimal environment, all messages are delivered. *Overhead* is the number of message copies for each message, which occurs due to message replication in the network. More message copies result in a higher delivery ratio but a high overhead results in more bandwidth and buffer consumption, which affects the distribution of other messages negatively. While applications using *Opportunistic Networks* are not (or should not be) time-critical because of the nature of these networks, these applications still desire a fast delivery and hence a low *end-to-end delay* of message delivery [3]. With our experiment, we want to answer the following questions: Can *opptain* achieve the preferred high delivery ratio? Can we accomplish a low overhead and a low *end-to-end delay*? Thus, we conclude this section with the most important question regarding the evaluation of our *Opportunistic Network*: How well does *opptain* perform?

Section Conclusion

To this point, we have formulated the problems that we have to face to establish an Android-based *Opportunistic Network*, and the resulting research questions and challenges. In the next section, we present our contributions that solve these research questions and challenges as well as resulting questions, challenges, and contributions.

1.3 Contributions

We documented the solutions and achievements of this dissertation in eleven publications. In the following, we first present the contributions that resolve the research questions and challenges from the last section and continue by presenting subsequent contributions.

The Creation of an Android-based Opportunistic Network

The main contribution of this dissertation is the *opptain* application which offers Android smartphones the possibility to participate as a node in a global *Opportunistic Network*. In Chapter 2, we present the solutions to the challenges of node discovery and connection establishment with the *Infrastructure Mode Wi-Fi* on Android smartphones, their identification in the network, the encrypted exchange of messages, the meta data exchange and message replication control, and show how to share the buffer between our network application and other apps on the smartphone. With these solutions, we have created an Android-based *Opportunistic Network* and can use this now for studying real-life *Opportunistic Networks*, and for evaluation, which we address in the next subsection.

Studies and Evaluation for our Android-based Opportunistic Network

After having built our Android-based *Opportunistic Network*, we want to evaluate it. To encourage others to use the application for our evaluation, we first study security threats and vulnerabilities to deploy a secure application and network. We present possible attacks on *Opportunistic Networks* and suggest solutions and defense mechanisms. Also, we investigate how many participants we need to establish a working *Opportunistic Network*. With the assumption that we need a rather dense than sparse network we first study node density in the simulator to use the results for our experiments. These experiments reveal how many devices are needed for a steady coverage and an efficient network. Lastly, we conduct the field experiment and show that our Android-based *Opportunistic Network* works as desired.

Improving One-Hop and Two-Hop Communication Options

By implementing *opptain* as a fully functioning *Opportunistic Network* on Android smartphones and evaluating it, we concluded the work on our initial research questions and challenges. During the evaluation we identified shortcomings, so, based on the solution we developed, we move on to new challenges and work out new contributions. For example, in our *opptain* application, we use the *Hotspotting* mechanism to connect devices, thus circumventing the limitations of the *Wi-Fi* network interface on Android smartphones that do not allow use of the ad hoc mode. With *Hotspotting*, the connection between two nodes is established by using the tethering hotspot on one device, which serves as the access point, so that other devices can connect to it as clients. An unconnected device scans its surroundings for hotspots and either connects to one it finds or becomes a hotspot itself if none are present.

We use the star topology that can result from *Hotspotting* to our advantage: While for small messages it can be useful and easy to route them over the hotspot node to another connected client, for large messages these two transmissions imply a lot of overhead. Forwarding the message directly to this other client can be more efficient. So, if a node needs to send a message to one of its hotspot's other clients, it can schedule a disconnect of both nodes from the hotspot to instead establish a direct connection to the more relevant node, thereby decreasing the effort to a single transmission of the message.

Another limitation we have to face is that Android's *Wi-Fi* network interface is not able to scan for other hotspots while in hotspot mode. The question that arises is if we can use other network interfaces, even on another personal device, to communicate with other nodes. We analyze additional signal ways for devices in Android-based *Opportunistic Networks* to answer the following questions: How can we communicate with other devices over a second signal way? How can we use smartwatches and other smart devices to extend our network interfaces? A possibility for the second signal way is the usage of smartwatches that are connected to the Android smartphones that are acting as nodes in the network. The smartwatch provides extra network interfaces that can examine the surroundings of the smartphone while the network interface of the smartphone itself is busy being a hotspot or a client. The challenge is to address the network interfaces on the smartwatch. Furthermore, we want to retrieve other information like sensor data or touch events from the smartwatch. We provide a library called *ansWEARs* for easy communication between smartphone and smartwatch that also offers the exchange of various information that the smartwatch is collecting about the network and its physical surroundings. This library is used to improve *opptain*, but can also be integrated into standalone Android projects.

Optimal Message Replication Control

One of the goals when designing *opptain* was the ability to include a variety of *routing protocols* ourselves and also to allow other developers to include their own *routing protocols* easily. This way, researchers, in addition to testing their *routing protocols* in a simulator, can also use *opptain* to evaluate them in the field. While working on *opptain*, we also experimented with *routing protocols* and thereby the replication control with the goal to improve flooding-based routing algorithms.

Routing in *Opportunistic Networks* has generally developed from simple flooding-based protocols like *Epidemic* to more sophisticated, utility-based protocols that rely on extra information about the network. Since there are not many implementations of real-world *Opportunistic Networks*, there are no clear instructions which protocol to use in which situations; therefore we can speculate that flooding-based *routing protocols* are particularly suitable for catastrophic situations. In contrast to the utility-based protocols, they do not require any information about the network that might actually not be available shortly after the establishment of the network due to the catastrophe at hand. However, flooding-based routing protocols use the device's resources extensively and imply a high overhead for the network. Thus, the questions arise: How can the overhead in the network be reduced when using flooding-based routing protocols? How can we be more resource-friendly?

We answer these questions extensively and improve the replication control in *Opportunistic Networks* in three different approaches, called *MOST-RPT*, *RPRS* and *ORBOPH*, that all vary the *forward strategy* and *drop policy*.

The basis of all three approaches is a formula for the delivery probability that is proposed with *MOST-RPT*, the first of the three replication schemes. Based on the current *hop* and *replication count* of a message, the formula estimates how likely it is for the message to be delivered to its destination. With the three replication schemes we present different proposals on how to employ this delivery probability. The *forward strategy* is the same in all three schemes: only messages with a high enough probability are forwarded, i.e. the messages whose probability exceeds a threshold. However, the schemes differ in the *queueing policy* and the *drop policy* which use *First-In-First-Out (FIFO)* or the ordering by delivery probability. In *RPRS*, we furthermore propose a *drop policy* that drops messages with high storage consumption and transmission costs first. With the last scheme, *ORBOPH*, we use information about the current network status to improve the replication control. While the first two schemes use a

dynamic threshold, *ORBOPH* calculates an optimal threshold based on the optimal *hop* and *replication count* of a message, which are estimated based on current network properties.

The presented evaluations of all three replication schemes show that these different combinations of policies and enhancements all perform better than the reference protocols.

Section Conclusion

In this section, we presented the contributions that solve our research questions and challenges as well as resulting questions and contributions. To conclude the introduction, we now illustrate the outline of this thesis before we continue to the detailed presentations and summaries of the papers.

1.4 Thesis Organization

In this chapter, we introduced and motivated our approach for an *Opportunistic Network* on Android smartphones. The solutions to the aforementioned research questions and challenges are documented in eleven publications that are presented as follows.

The subsequent chapters 2 to 5 provide summaries and contributions of our publications and state their impact on this thesis. In Chapter 2, we present the concept and implementation of our Android-based *Opportunistic Network*. First, we introduce our *opptain* application [4] in Section 2.1. We further discuss the decisions which were made for *opptain* and which lay the foundations for this dissertation. Other challenges arising from these decisions are solved in the other publications. In Section 2.2, we present our solutions for our challenges about time and buffer management [5]. In Chapter 4, we improve the one- and two-hop communication in our *opptain* network. In Section 4.1, we utilize the star topology of the *Hotspotting* mechanism by scheduling promising future contacts [6]. In Section 4.2, we present our solution for wearable devices, especially smartwatches, as second signal way for *opptain* [7]. In Chapter 3, we conduct studies and evaluations regarding our *opptain* network. In Section 3.1, we describe a study on the impact of node density and transmission range in *Opportunistic Networks* [8]. In Section 3.2, we discuss security threats and mitigation techniques in Android-based *Opportunistic Networks* [9]. Closing, in Section 3.3, we provide a field experiment and its evaluation [10]. In Chapter 5, we present three solutions to improve flooding-based *routing protocols*: *MOST-*

RPT [11] in Section 5.1, *RPRS* [12] in Section 5.2 and *ORBOPH* [13] in Section 5.3. Finally, in Chapter 6, we conclude our work and look into future work.

Chapter 2

Android-based Opportunistic Networks

In this chapter, we address the challenge of applying *Opportunistic Networks* to smartphones and lay the foundation for this thesis. To establish and optimize an *Opportunistic Network*, regardless of the deployment on smartphones or other devices, we basically need to provide the following two functionalities: Nodes can establish connections and nodes can exchange messages. To enable the connection establishment, we require a mechanism for node discovery. Furthermore, the message exchange, in turn, consists of several sub-challenges. The *routing protocol* dictates what meta information should be exchanged and used for forwarding and replication. The *forward strategy* includes the *queueing policy* which determines the order of the forwarding queue. If the buffer is full, we need a *drop policy* that chooses which messages to remove from the buffer first [3].

The goal of this dissertation is to address the aforementioned challenges and establish a smartphone-based *Opportunistic Network*. In this chapter, we start with the neighbor discovery and connection establishment, the exchange of meta information and the forwarding and replication of messages based on *routing protocols*. From related work, we know about the multitude of *routing protocols* for *Opportunistic Networks*, but all these protocols have mainly been tested in simulators, partly due to the fact that there is no established real-world *Opportunistic Network* to test these protocols with. As smartphones are the mobile devices that are most widely spread, they can provide such a real-world testbed when equipped with our software. To evaluate multiple routing protocols for multiple application situations, one demand for our work is to include a high variety of *routing protocols* in our network application and to give future developers the possibility to include new protocols. A detailed description of *routing protocols* can be found in our paper [4] or in Chapter 5 in which we will address replication schemes for *routing protocols*.

The most rudimentary demand for nodes in *Opportunistic Networks* is the feasibility to establish connections to each other. These connections should be set up automatically and not require any user interaction. Another important demand is that the smartphones should not be *rooted* so that the devices can be used off-the-shelf, which increases the acceptance and spreading of our application, especially in catastrophic situations in which *rooting* is a bigger challenge.

We have to choose an operating system and a connection technology which meet these demands. While working on this thesis, only two operating systems are widespread enough to be considered, Android and iOS [2]. When looking into both operating systems, we notice that neither supports an ad-hoc-mode for *Wi-Fi*, and *Bluetooth* demands user interaction at each connection (in early versions of Android and overall in iOS). Other wireless ad-hoc communication standards like *LTE Direct*, *NFC* or others are not supported by current devices or are not usable due to their short range or once again necessary user interaction. However, *Infrastructure Mode Wi-Fi* is available on Android smartphones and can be used without user interaction. Thus, we choose smartphones running the Android operating system as nodes for our *Opportunistic Network*. Choosing Android over iOS yields another advantage: With a market share of about 86%, Android is more widespread [2]. Another advantage of using *Wi-Fi* infrastructure mode on Android is that we use a technology which is not Android-specific; this means that other devices, even non-smartphones, can participate in the network created by the Android devices.

Following this introduction, we first discuss related work. Afterwards, we introduce the first version of our network application *opptain* in Section 2.1 and take care of time and buffer management in Section 2.2. With the ready to use *opptain* application that results from this chapter, we can pursue our research in multiple ways. In Chapter 3, we reflect by studying and evaluating our application. In Chapter 4, we attempt to improve the meta data and message exchange in *communication islands* by extending the neighbor discovery and connection establishment. In Chapter 5, we create multiple replication schemes which can be used for inclusion in our *opptain* application.

Related Work on the Spontaneous Connection of Android Smartphones

In this section, we list related work about attempts to establish spontaneous connections between smartphones which were made before or in parallel to this thesis. The demands for our Android-based *Opportunistic Network* are the automatic connection establishment of *unrooted* Android smartphones without user interaction and with a high transmission range. First, we provide a list of conventional connection technologies that are available to Android smartphones and state why we do not use them.

Conventional connection technologies that are available on smartphones and can be used for spontaneous connections are *Bluetooth*, *Bluetooth Low Energy*, *NFC*, *LTE Direct*, and *Wi-Fi*. At the time that we started our work, Android and other operating systems did not allow the connection via *Bluetooth* without pairing the devices. Because the pairing of the devices was only possible with user interaction, we could not use *Bluetooth*. *Bluetooth Low Energy* is a battery saving alternative to *Bluetooth*, which was designed for connecting devices that are in close range. Apart from *Bluetooth Low Energy* not being considered as it was not available on smartphones when we started this work, it would not fit the demands for our *Opportunistic Network* because of its small transmission range. Since *NFC* has an even shorter transmission range than *Bluetooth Low Energy*, it does not meet our demands either. *LTE Direct* has a high transmission range but is not yet available for Android and other operating systems. Consequently, we use *Wi-Fi* as the connection technology for our Android-based *Opportunistic Network* and use the aforementioned *Infrastructure Mode Wi-Fi* to connect smartphones.

Next, we provide a list of approaches that have been developed before or in parallel to our work and state the differences to our *opptain* application. These approaches contain solutions with regard to the connection possibilities of Android smartphones, solutions that use mesh networking, and solutions that either require *rooted* smartphones or user interaction.
Thinktube Inc. developed and implemented *ad-hoc* mode support for Android versions between 4.2.2 and 5.0 [14]. The modifications were included into both the Android Open Source Project version [15] and the modified Android version by Cyanogen Inc. There are currently only five devices that are able to use the Thinktube Inc. ad hoc mode; the Samsung devices Galaxy S, Epig 4G, Galaxy Nexus, Nexus S, and the Asus device Nexus 7 (with the Nexus devices being produced by Google and only manufactured by Samsung and Asus). We, however, want to create a solution for an Android-based *Opportunistic Network* that can run on all Android devices and not only a few. Simultaneous to our work, Trifunovic et al. created WLAN-Opp [16] which

uses the same *Hotspotting* mechanism as *opptain*. The work focuses only on the connections of the smartphones and not on the messaging and routing, and uses open access points in their field test. Our solution *opptain* is a complete framework that not only establishes connections between devices but also offers a possibility to include and test a variety of *routing protocols* and to connect third-party user applications. The Serval Project [17], financed by the Shuttleworth Foundation, is an Australian project that aims to connect smartphones through *Wi-Fi*. In contrast to *opptain*, the Android application of the *Serval Mesh* project requires *rooted* smartphones and uses mesh networking instead of opportunistic networking. Another open-source software for Android devices that in contrast to *opptain* uses mesh networking to route messages in the network is Briar, which offers device connections without a centralized server. Briar is promoted as a "messaging app designed for activists, journalists, and anyone else who needs a safe, easy and robust way to communicate" [18]. Messages are synced via Internet if available and via *Bluetooth* or *Wi-Fi* if there is no connection to the Internet. With *opptain*, we want to fully adapt the principles of *Opportunistic Networking* and not only reduce the connections to mesh networking. Further examples for mesh networking on Android smartphones that are however proprietary and therefore not extendable to our objective are Open Garden's MeshKit SDK and FireChat [19], which was used in civil protests, and *Bridgefy* [20], which offers an SDK to connect Android devices to each other. Johannes Morgenroth et al. present IBR-DTN for Android devices [21]. It is an implementation of the Bundle Protocol [22], but does not offer the connection between devices and instead relies on an open *Wi-Fi* access point which the smartphones are connected to.

For our goal, to establish a real-world, smartphone-based *Opportunistic Network* as an application and a test framework for developers, we have to develop our own approach which is presented in the next section.

2.1 Infrastructure Mode Based Opportunistic Networks on Android Devices

This section summarizes the contributions of our papers [23] and [4].

Andre Ippisch and Kalman Graffi.
"An Android Framework for Opportunistic Wireless Mesh Networking".
In: *Proceedings of the GI/ITG International Conference on Networked Systems (NetSys)*. 2015.

Andre Ippisch and Kalman Graffi.
"Infrastructure Mode Based Opportunistic Networks on Android Devices".
In: *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA)*. 2017. Acceptance Rate: 29%.

We present the contribution of the second paper [4] in Section 2.1.1 and the importance and impact on this thesis in Section 2.1.2. We summarize the paper in Section 2.1.3 and declare the personal contribution in Section 2.1.4. The verbatim copy of our paper [23] can be found on page 0 and the verbatim copy of our paper [4] can be found on page 0.

2.1.1 Contribution

The contribution of this paper is a mobile application called *opptain* which establishes an *Opportunistic Network* on Android devices. In this paper, we introduce the Android-based *Opportunistic Network* which is our solution to the central research question of this thesis. The goal is to enable *Opportunistic Networks* as an everyday help, not only in theory but also in practice. This paper results in the implementation of the demands and the solutions to the challenges that arise from some of these demands. The demands that have been implemented are, amongst others, running the application on off-the-shelf unrooted Android smartphones in the background and without user interaction. We provide the network application, an *API* and some user applications for the application layer of the network. These user applications are a messaging app and a filesharing app which we provide as proof of work. We present test results for the link connection layer and proof of work for the routing possibilities and the user applications.

2.1.2 Importance and Impact on Thesis

The application *opptain* presented in this paper is the core of the dissertation. It marks the beginning of research for this work and all contributions are based on or motivated by this application.

2.1.3 Paper Summary

In this paper, we motivate and introduce Android-based *Opportunistic Networks*. Nowadays, we have smartphones that produce and store large files that are not shared with geographically close users directly but over the Internet. Disadvantages are the transmission speed over the mobile network, the limited and costly data plan, or the possible absence of mobile infrastructure and hence the missing possibility of using the Internet to share data. Our first goal is to transmit files of any size in a secure, fast and straightforward fashion from one smartphone to another. Smartphones and their users form an *Opportunistic Network* by communicating over *Wi-Fi* and without mobile network infrastructure. The question that arises is how to connect the devices to each other since ad-hoc connections over *Wi-Fi* are not available on current smartphones. The application must run on off-the-shelf smartphones, perform all operations in the background and without user interaction, connect the devices, and transfer data securely. With this paper, we offer an Android application called *opptain* that connects over *Wi-Fi* and with high transmission speed identifies devices, encrypts messages, and uses multiple *routing protocols* to deliver the message to its destination.

One of the main challenges of this paper is connecting the Android devices automatically, without user interaction and with high-speed transmission. The ad hoc standards that are implemented on Android smartphones do not fulfill these demands. So, we use the tethering hotspot on one side as a server and on the other side connect to it as a client. This takes place in an automated fashion called *Hotspotting*, in which each device scans for an available hotspot and either connects to one it finds or starts acting as hotspot itself if there are none. Even devices that have no possibility to open a tethering hotspot can be included in the process when acting only as a client. For the identification of the devices and the encryption of the messages, we create a private/public key pair. The fingerprint can be used as identification, the keys for authentication and encryption inside the network. Instead of focusing on one *routing protocols* only, we try to gather and offer as many meta information as possible about the network to be able to apply multiple routing protocols. For this paper, we focus on meta information exchange over two hops, so that nodes know how to route messages to another

node that is up to two hops away. In addition to the network application *opptain*, we provide an *API* so that user applications can use the *opptain* network for message exchange. Examples for these user applications and proof of work for our proposal are a filesharing and a messaging application which we provide as apps for distribution along *opptain*.

One of the main goals for this work is the adaption of our *opptain* network by many smartphone users. First, we verify that *opptain* operates as desired; smartphones connect to other smartphones in turn and exchange data opportunistically. Next, we test multiple aspects of the network application that will be most important to the users. These tests cover battery life, transmission speed and range, the speed of encryption and decryption and the connection and routing information. We can see that without other apps running, our devices still run for a day. Transmission's speed is an improvement to most mobile data connections and the range is adequate for a mobile *Opportunistic Network*. We show that the encryption of data is fast enough to fully use the transmission stream and that routing information is distributed correctly in the network.

2.1.4 Personal Contribution

The 31st IEEE International Conference on Advanced Information Networking and Applications (AINA 2017) accepted 163 submissions with an acceptance rate of about 29%. The paper received only reviews with a "Must Accept" recommendation for an outstanding submission.

Andre Ippisch is the main contributor and author of this paper, Kalman Graffi contributed to the methodology of the research and the revision of the paper.

2.2 Time and Space in Android-based Opportunistic Networks

This section summarizes the contributions of our paper [5].

Andre Ippisch, Tobias Küper and Kalman Graffi.
"Time and Space in Android-based Opportunistic Networks".
In: *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA)*. 2018. Acceptance Rate: 28%.

We present the contribution of our paper in Section 2.2.1 and the importance and impact on this thesis in Section 2.2.2. We summarize the paper in Section 2.2.3 and declare the personal contribution in Section 2.2.5. The verbatim copy of our paper [5] can be found on page 0.

2.2.1 Contribution

This paper resolves the challenges for our network application *opptain* in competing against other applications and in synchronizing devices, which arise as we introduce the possibility of *Opportunistic Networks* for smartphones. In this paper, we propose *drop policies* for a reasonable buffer management in shared storages to ensure that no other data of the user is impaired. Our policies see to it that the buffer is diminished by deleting messages in time before any personal data gets discarded. Instead of creating copies, we use references when sending bundles to avoid duplicated data and unnecessary storage usage. Additionally, we provide a technique to handle a message's *Time-To-Live* on different devices and to establish global event ordering without any synchronization and in response to time changes.

2.2.2 Importance and Impact on Thesis

Many applied *Opportunistic Networks* [24, 25] use hardware that provides only one function and is precisely constructed for the task at hand. Albeit our Android-based network only has one task as well, it runs on a device serving multiple purposes, as the user purchasing the smartphone has a different intention. Although in its focus are traditional features like calls and text messages, today there are a lot more apps installed on every smartphone. This leads to the struggle for storage space between multiple applications and also between apps and the

user. A buffer is considered part of every *Opportunistic Network*, however, our application has to pay heed to more important data on a device, data that is created by other apps. When connecting devices with a mobile network or the Internet, this facilitates clock synchronization, whereas *Opportunistic Networks* lack this ability because there is no central instance. Without a clock synchronization between devices, unsynchronized clocks can result in issues regarding the *Time-To-Lives* and the succession of messages at all nodes. These issues of buffer and time management demand solutions, which were found and integrated into our application *opptain*.

2.2.3 Paper Summary

In this paper, we present our extensions to our first version of *opptain* which keeps the two challenges of time and buffer management open.

Every message that is transferred through an *Opportunistic Network* has a *Time-To-Live*. This *Time-To-Live* indicates the point of time at which the message should not be further replicated but discarded; a message, for example, created at 10:00am with a *Time-To-Live* of 30 minutes will be removed from any buffer at 10:30am. The handling of this time specification is associated with the clock of the devices. For tracking the *Time-To-Live* and keeping the messages in the correct order, the devices have to keep track of not only their own clock, but the clocks of the other devices as well. Clock synchronization would sort out this problem, as all devices would have the same clock. However, *Opportunistic Networks* have no central instance which can be used to synchronize time between devices. While in centralized networks, clocks can be synchronized with the central instance, for example a server or in case of the Internet *NTP*, in decentralized networks clock synchronization is not that trivial. While there are examples of how to synchronize the clocks in decentralized networks, we have to take a look at Android to see if they are suitable.

Regarding buffer management, we need to deal with the shared storage between all apps on the smartphone. The Android storage that our network app shares with all other apps must never be full or near full because if it is, applications might not work properly. While our application's message buffer should take the storage space it can get to serve the purposes of our software, it should not disturb the user's everyday workflow by overwriting any data or cluttering up the storage. This is why we have to delete messages from the network-specific part of the storage if either the network buffer or the other apps need space. Android lacks a reliable warning system indicating that the storage is running full, thus to prevent the storage from being crammed, we

have to detect a full storage ourselves. For the decision which messages to keep or remove, we have to use appropriate forward and drop policies. Additionally, messages should be clustered into different storage locations depending on the role that the node plays for these messages, i.e., source node, relay node or destination node, to differentiate when messages have to be dropped.

For related work, we present clock synchronization mechanisms under the assumption that the Internet and hence *NTP* are not available. Also, we comment on *GPS* which cannot be used for clock synchronization in Android. Concluding, we need to find a possibility to keep track of time for messages. For the buffer management, we present related work on *drop policies* which are all applied to a one-buffer system. We need to find a *drop policy* that can be utilized on a device that runs not only our *Opportunistic Network* but also other user applications whose storage must not be altered.

The solutions of this paper, like the challenges, are twofold; first we have the time-related challenges, and second the space-related challenges. As no related work can be used to synchronize the clocks between devices and our network application cannot induce a time change on the device even if it learns the clock of another device, we are not able to employ clock synchronization in our *Opportunistic Network*. Thus, we need another solution to guarantee that the *Time-To-Live* of a message is handled correctly on every device. The *Time-To-Live* as well as the time that a message was sent are stored in the message's header. There are two options for storing the *Time-To-Live*: Absolute or relative. The relative *Time-To-Live* would be dynamic and indicate how much time is left. The absolute *Time-To-Live* would never change and just state the overall time for which a message should be in the network. They can both be applied, however we choose to use the absolute *Time-To-Live*, as we wanted the original message to not be edited to keep the signature of the message and the original message header. Instead, we add new information to the non-signed extended header by storing the offset between devices. As each receiver knows the offset between his and the sender's clock, it can combine this known offset and the saved time difference to recalculate the message's correct *Time-To-Live* or creation time. Additionally to the time differences between devices, our application has to cope with clock changes on a single device, following which all messages in the buffer have to be adapted. Android does notify apps about a time change but not about the actual difference between the old and new time. We identify this difference by taking the time since the device's last reboot and using this time span to calculate the difference to the new current time by which the offset of the messages has to be adapted.

We continue with the space-related challenges. First, the storage options on Android devices are described. There are internal, external and emulated storage and these can be divided into both private/public and persistent/cached storage. We assign different roles to our network nodes: originator, relay, or one of multiple destinations. For each role, we define where to save specific messages and when to drop those if the buffer space is running low. For different scenarios and different devices, we test when the buffer is running full and how Android apps are notified about low buffer space. We can see that most devices are notified as soon as the buffer is filled to about 90 to 100 percent; however, there are devices that can run full to 100 percent occupied buffer and which leads to errors on the phone. To prevent these errors we present how *opptain* checks the storage's current capacity periodically. For the decision which messages to keep or remove, we give appropriate *drop policies*.

For the time-related issues, the reaction on time changes and the time synchronization is solved, and proof of work is given by field tests. Messages have to contain information about the time difference between devices so that user applications can put messages in context. The shared buffer space challenges are solved by using appropriate storage options, observing free buffer space and reacting accordingly with specific drop policies. The evaluation shows that for all test devices the buffer management works as expected.

2.2.4 Related Work on Time and Buffer Management

In this section, we want to present related work in regard to the topics of clock synchronization and buffer management. For clock synchronization, we first have to discard any options that rely on a centralized instance, for example *NTP* [26] or the synchronization via the mobile cell tower. While these options are the most favorable for devices that are connected with the Internet or at least with a cell tower, we cannot rely on a centralized instance in our network that is completely decentralized. Another system that is kind of centralized but available via long distance transmission is *GPS* [27]. *GPS* data is used by devices to find their location on Earth but can also be used to synchronize their clocks as the time is sent with the location information. While *GPS* data is available for our smartphones, they are not using *GPS* to synchronize their clock and with off-the-shelf Android smartphones, it is not possible to include this functionality. This brings us to the point in which even solutions for decentralized networks [28, 29, 30, 31] that would be possible to implement on hardware devices have no gain for our *opptain* application since we cannot change the time on our device.

For the topic of buffer management in *Opportunistic Networks*, there is a variety of work as well as summaries and evaluation available [32, 3, 33]. The *drop policies*, however, are always assumed to be applied to a hardware storage that is completely assigned the buffer of the node in the *Opportunistic Network*. For our work, we want to apply our principles of *Opportunistic Networking* to smartphones that humans also use for many other tasks. Therefore, our network application has to share its buffer with the buffer space of other applications on the smartphone; a problem for which there is no related work known to us.

2.2.5 Personal Contribution

The 32nd IEEE International Conference on Advanced Information Networking and Applications (AINA 2018) accepted 157 submissions with an Acceptance Rate of about 28%.

The author of this dissertation, Andre Ippisch, had the idea for this paper and developed the algorithms. Further contributions of Andre Ippisch are the conception of the research, the drafting of the paper, and the joint evaluation of the solution. The implementation and the joint evaluation of the solution were conducted by Tobias Küper. Andre Ippisch and Tobias Küper contributed to the writing and discussion of the paper. Kalman Graffi contributed to the methodology of the research as well as to the revision of the paper.

Chapter 3

Studies and Evaluation for our Android-based Opportunistic Network

In this chapter, we address the challenges regarding the analysis and evaluation of the previous contributions. Now that we have solved the challenges of node discovery, connection establishment, identification, encryption, and time and buffer management for our *opptain* application, we want to evaluate this system in an open field test to see how well it does. Prior to the open field test we conduct two studies to enhance the premises of the field experiment.

First, we identify how many devices and hence participants we need for a meaningful test. This is expressed by the node density of the network, whereat we assume it has to be high. Only if there are enough nodes incorporated in the network inside the test area, messages can be carried from node to node and, thus, from source to destination. In Section 3.1, we present a study [8] on the influence of node density and transmission range on the contact probability in *Opportunistic Networks* to answer the question of a suitable node density for tests of our *opptain* network.

Next, we want to make sure that our application is secure in regards to software- and network-based threats before we start the open field test. So far, we only incorporated security with the end-to-end encryption for messages. To encourage others to use the application as participants in our experiment, we want to be sure that there are no vulnerabilities. In Section 3.2, we present a study [9] on the possible security threats and their solutions or mitigation techniques for our Android-based *Opportunistic Network opptain*.

Finally, with these results, we know how many nodes we need to establish a functional *opptain* network and have to find a group of people and a bounded area to conduct a field experiment with significant results. In our experiment we analyze the average *overhead*, *delivery ratio*, and *end-to-end delay* in our network with different *routing protocols* and *Time-To-Lives*. In Section 3.3, we present the realization of this field experiment [10] of our *opptain* application and its results.

3.1 The Impact of Node Density and Transmission Range on Opportunistic Networks

This section summarizes the contributions of our paper [8].

Andre Ippisch, Salem Sati and Kalman Graffi.
"Device to Device Communication in Mobile Delay Tolerant Networks".
In: *Proceedings of the International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)*. 2017. Acceptance Rate: 46%.

We present the contribution of our paper in Section 3.1.1 and the importance and impact on this thesis in Section 3.1.2. We summarize the paper in Section 3.1.3 and declare the personal contribution in Section 3.1.4. The verbatim copy of our paper [8] can be found on page 0.

3.1.1 Contribution

To forward messages to other nodes in *Opportunistic Networks*, there have to be nodes in the surroundings that the messages can be transferred to. This assumes that there are enough participating nodes in the area and a steady coverage is achieved. The contribution of this paper is a study on the influence of node density and transmission range on contact probability in *Opportunistic Networks*. We evaluate the effects of different node densities in contrary to previous research which only evaluates fixed node density in areas with a fixed size. We present a mathematical model for the mobility of the nodes, the Inter-Contact Rate, and the contact probability. We provide experiments and an evaluation of the influence of node density-related parameters on each other and on the system performance. The results can be used to reduce the amount of overhead and to improve the efficiency of flooding-based protocols like *Epidemic*.

3.1.2 Importance and Impact on Thesis

As long as Android-based *Opportunistic Networks* are not established globally, the *opptain* network is more sparse than dense in relation to Earth as the network area. In small groups, however, in which *opptain* is used, for example in an office scenario, the network is considered

to be more dense than sparse. Information about the effects of node density in *Opportunistic Networks* is essential for this work to be prepared for a variety of application scenarios. And since *opptain* can be used with multiple wireless technologies like *Wi-Fi* and *Bluetooth*, different transmission ranges have to be considered in a simulation analysis, too.

While working on multiple replication schemes for *Opportunistic Networks* (see Chapter 5), for evaluation, we always used a fixed number of nodes in a bounded area. This was the approach of most of the researchers in this field and for comparison purposes, it seemed to make sense to continue this approach. For the topic of real-world, Android-based *Opportunistic Networks*, namely *opptain*, it was, however, necessary, to search for work on the impact of node density in *Opportunistic Networks* and *Delay Tolerant Networks* in general. Because the work in this field is limited, again, mostly because of comparison purposes, we investigated this node density impact ourselves to get results for this thesis.

3.1.3 Paper Summary

In this paper, we evaluate the *Epidemic* routing protocol with a focus on node density in our network. Before we can run a field experiment to analyze the efficiency of the network that *opptain* establishes, we want to find out how many devices and hence participants we need for a meaningful test. This is expressed by the node density of the network. For our purpose, as the number of connections is an indication for the node density, we define node density in terms of average node connectivity. In *Opportunistic Networks*, node density fluctuates constantly. One factor for node density changes are the mobility patterns of different node groups. In such networks with isolated islands of nodes, *Store-Carry-Forward* is used by routing protocols to disseminate messages. *Opportunistic Networks* with changing node densities induce a lower contact probability which affects the routing performance negatively. The presented previous research focuses on the message dissemination process without considering the density of nodes, whereas we feature results for node density as a function of the number of nodes and their transmission range. We investigate the impact that node density and transmission range have on the dissemination speed of messages, the contact probability, and the energy consumption. We want to investigate a wide spectrum of node density. Both extremes bear problems: Very few nodes limit the connection possibilities and the network will probably fail. The other way around, too many nodes in one place, for instance during a concert, could overburden the network. Routers are already struggling with high amounts of users in close-range, so smartphones with their less powerful network interface cannot handle hundreds of simultaneous connections. Everything between those extreme cases should result in better outcomes.

To evaluate the node density impact, we first define our system model for mathematical analysis based on models from previous research [34, 35, 36, 37]. With this model, we can investigate the impact of node density or rather of various node density parameters and different transmission ranges on numerous aspects of our network and the message transmission process. The model also considers different mobility patterns to simulate different traffic situations. First, we investigate the impact of node density and transmission range on the *Inter-Contact Time*, which we assume is exponentially distributed. Based on the transmission range and the node density, we provide equations to calculate the *Inter-Contact Time* for the Random Direction mobility model, the Random Waypoint model and for our own model, which is a combination of the Shortest Path and Random Route mobility models [38]. From our evaluation, we can see that the *Inter-Contact Time* is not impacted by the node density.
Second, we investigate the impact of node density on the contact probability and the number of contacts per hour. Therefore, we had to find a manner of calculating the contact probability in terms of node density and transmission range. We define a ratio and threshold which determine if a network is dense or sparse and a node density equation for the calculation of the surrounding density of a specific node. We confirm that networks with higher density and higher transmission range have an increased contact number. The contact probability is almost equal for different densities but increases with higher transmission ranges.
Third, we investigate the impact of node density on different routing metrics, namely *end-to-end delay*, the amount of relayed messages, and *delivery ratio*. We evaluate the impact of node density by comparing Epidemic with different numbers of nodes and transmission ranges. We can observe that the delivery ratio decreases for a higher number of nodes and stays almost equal for different transmission ranges. This is because the delivery probability is increased when the dissemination speed and message drop rate decrease, which is the case with a lower transmission range. The number of relayed messages, however, increases with a higher number of nodes and transmission range because of the higher dissemination speed and more buffer space in the whole network.
Last, we investigate the impact of node density on the energy consumption in the network. We consider *Bluetooth* and *Wi-Fi* as interfaces and the *Internet Protocol Neighbor Discovery (IPND)* protocol which was developed to discover existence and availability of encountered nodes. We use different intervals of the *IPND* protocol to evaluate the energy consumption. For all different beacon intervals, the energy consumption is decreasing for a higher number of nodes and higher transmission ranges.

For the experiments, three scenarios with different node densities are presented, which consist of 66, 126 and 246 nodes and transmission ranges from 50 to 250 meters. *The ONE simulator* and the Helsinki map are used for conducting the experiments. We conclude that we can

increase dissemination speed with a higher node density but with the cost of higher resource consumption.

3.1.4 Personal Contribution

The reviewers of the International Symposium on Distributed Simulation and Real-Time Applications (DS-RT 2017) recommended the paper for the top 30% of all submissions. 50 papers have been submitted and 23 have been accepted for publication which results in an acceptance rate of 46%.

The joint drafting of the methodology, solution and paper, as well as the initial development of the mathematical analysis part of the paper and the conduction of the experiments and their evaluation and analysis was done by Andre Ippisch and Salem Sati. Andre Ippisch conducted all necessary implementations and Salem Sati did the details of the mathematical analysis part of the paper. Kalman Graffi contributed to the methodology of the research as well as to the revision of the paper.

3.2 Mitigation Techniques for Software- and Network-based Threats

This section summarizes the contributions of our paper [9].

Andre Ippisch, Martin Nowak and Kalman Graffi.
"Mitigation Techniques for Software- and Network-based Threats in Android-based Opportunistic Networks".
Technical Report: TR-2018-002. Technology of Social Networks Group, Heinrich Heine University, Düsseldorf, Germany. 2018.

We present the contribution of our paper in Section 3.2.1 and the importance and impact on this thesis in Section 3.2.2. We summarize the paper in Section 3.2.3 and declare the personal contribution in Section 3.2.4. The verbatim copy of our paper [9] can be found on page 0.

3.2.1 Contribution

The contribution of this paper is a summary of attacks, vulnerabilities, and threats against but also solutions and mitigation techniques for *Opportunistic Networks* on Android devices, specifically for our Android-based *Opportunistic Networks opptain*. In this work, we study related scientific literature and analyze our network application *opptain* to cover not only a large base but also detailed aspects of the security in Android-based *Opportunistic Networks*. We do this to offer a secure application and to mitigate threats against the network the application creates. We have to secure different parts of the network; first, the communication between the network application and user applications on the Android device, second, the communication between devices. We implement proof of work applications to show the discovered attacks and their mitigation techniques. These examples can be helpful guidelines for developers of secure Android-based *Opportunistic Networks* or Android projects that involve direct communication between devices.

3.2.2 Importance and Impact on Thesis

This paper covers the very essential topic of security in Android-based *Opportunistic Networks*. The *opptain* project started as a proof of work project which covered the basics of *Opportunistic Networks*. We started early to include common security aspects into the application, also based on best knowledge. Messages in the network are encrypted to offer secure end-to-end encryption. Furthermore, we fixed the random number generator of Android, as it only offers a slightly broken generator that has to be repaired by developers. After the groundwork was done, a work on security threats was conducted, to find issues in the system and to remove or mitigate them.

3.2.3 Paper Summary

In this paper, we study software- and network-based threats against our *opptain* application and try to find solutions or at least mitigation techniques for those threats. By this we want to make sure that our application is secure in regards to software- and network-based threats before we start the open field test. We first categorize the attacks as passive or active; with passive attacks an attacker tries to gather information without the attackees knowing and with active attacks the attacker obviously disrupts the normal functionality. Desired characteristics of a secure and reliable *opptain* network are availability, confidentiality, integrity and authentication.

First, we focus on vulnerabilities and attacks that target the Android device and *opptain*, for which we want to find mitigation mechanisms. We do not cover attacks on *rooted* devices since adversaries on rooted devices have non-restricted access to all data. We cover a multitude of attacks and vulnerabilities that can be carried out by malicious applications, for example Intent-*based Attacks*, *App Repackaging*, *Denial of Service*, *Binder Attacks*, and *User Interface-based Attacks*. Intents are used for intra- or inter-application communication on Android devices; with Intent-*based Attacks*, *opptain* messages become accessible by third-party applications, and applications can access components through a more privileged application. By *repackaging* the install file of an application, malicious code like Trojan horses or malware can be inserted into the system. *Denial of Service* attacks can terminate applications and force a reboot of the system. Android applications are sandboxed and use so called *Binders* for inter-process communication, which can be used to inject faulty transactions into services and bypass sanity checks. *User Interface-based Attacks* include the hijacking of tasks in Android's multitasking mechanism, or clickjacking where an opaque user interface covers the attacked app to hijack touchscreen clicks.

Second, we cover attacks against routing and communication between Android devices that are participating in the *opptain* network. Contrary to before, in these scenarios an attacker has full control of the device. Attacks and threats covered in this paper include *Blackhole Attacks*, *Hotspot Spoofing*, *Sybil Attacks*, and *Denial of Service*. *Blackholes* are malicious nodes that attract messages and drop them instead of forwarding them. *Hotspot Spoofing* is an attack in which a malicious node impersonates the identity of another node. In a *Sybil Attack*, an attacker creates fake identities, possibly also by impersonating multiple other nodes. *Denial of Service* attacks in this category are conducted by flooding nodes with many messages.

Next, we present solutions and mitigation techniques for the attacks, vulnerabilities and threats of both categories mentioned before; on one device, and between multiple devices. We only use solutions that do not require changing system components and can be directly implemented into an application. Regarding attacks on one device, we present mechanisms against Intent*-based Attacks*, and provide *Access Control*, *Binder Attack Mitigation*, and an Intent *and Message Threshold* as solutions and mitigation techniques. To stop Intent*-based Attacks*, for example, we use only explicit *Intents*, which means that these Android messages have an explicit receiver inside the Android system. Also, with *Access Control*, we provide a mechanism that handles all interactions between *opptain* and any third-party applications to allow only participating apps to communicate with *opptain*. We can use this as a solution for the Intent*-based Attacks* and for *Binder Attacks*. With an Intent *and Message Threshold*, we can stop *Denial of Service* on the device by limiting the amount of *Intents* and messages that are accepted. Mitigations against *App Repackaging* cannot be implemented into an application since the application itself is what is changed by an attacker. The integrity of the application can hardly be verified by the Android device or the user without a centralized instance in the network. Also, *User Interface-based Attacks* rely on the smartphone user granting malicious apps the permission to cover other applications; this is a permission that users should never give to apps they do not trust. There is no possibility to implement a mitigation technique into our applications that prevents this behavior.

For attacks against routing and communication between Android devices, like *Blackhole Attacks* or *Sybil Attacks*, or to stop *Denial of Service* between multiple devices, we propose a form of social trust in the network and discuss it. We can, for example, apply *Trust and Reputation* metrics and *White- and Blacklisting* to decide if messages should be forwarded to a node. To tackle *Hotspot Spoofing*, clients can check the fingerprint of the public key of each hotspot to verify the identity and terminate the connection if there is a mismatch. Finally, by *enhancing messages*, we encrypt the message body and sign the whole message to stop message manipulations and guarantee confidentiality, authenticity and integrity for all initial message

attributes. We further discuss all solutions and mitigation techniques and also provide example implementations for these techniques as well as for attacks and vulnerabilities.

3.2.4 Personal Contribution

The author of this dissertation, Andre Ippisch, had the idea for this paper and contributed to the joint conception of the research. The investigation, implementation, and evaluation of the solution were conducted by Martin Nowak, as well as the joint conception of the research. Andre Ippisch and Martin Nowak contributed to the writing and discussion of the paper. Kalman Graffi contributed to the methodology of the research as well as to the revision of the paper.

3.3 Field Experiment on the Performance of an Android-based Opportunistic Network

This section summarizes the contributions of our paper [10].

Andre Ippisch, Philipp Brühn and Kalman Graffi.
"Field Experiment on the Performance of an Android-based Opportunistic Network".
In: *Proceedings of the International Conference on Parallel and Distributed Computing (Euro-Par)*. 2018. Acceptance Rate: 29.4%.

We present the contribution of our paper in Section 3.3.1 and the importance and impact on this thesis in Section 3.3.2. We summarize the paper in Section 3.3.3 and declare the personal contribution in Section 3.3.4. The verbatim copy of our paper [10] can be found on page 0.

3.3.1 Contribution

For this paper, we conducted field experiments for our Android-based *Opportunistic Network opptain* and present the results. The quality of *Opportunistic Networking* applications is assessed based on the three quality measures *delivery ratio*, *overhead*, and *end-to-end delay*, hence we execute tests to analyze how well *opptain* is doing in regard to these measures. To conduct these experiments, we developed a test framework called *oggregator*. With this test framework, we can, also opportunistically, distribute the settings files for test runs to all test devices. After one or multiple test runs, which start automatically, each device's result data files are aggregated on one device for uncomplicated evaluation. We also provide a methodology for evaluating *Opportunistic Networks* consisting of the data to collect while testing, the metrics for evaluation, and the setup of the field test. We present performance results for real-world *Opportunistic Networks* that show that *opptain* provides robust performance for our user applications, for example for transmitting and delivering chat messages or files.

3.3.2 Importance and Impact on Thesis

One of the challenges of this thesis is the design, implementation, and evaluation of real-world Android-based *Opportunistic Networks*. The position paper for *opptain* (see Section 2.1)

summarizes the design and implementation of this overall research and presents a proof of work as well as an evaluation. That evaluation, however, focuses on different metrics which only focus on a limited view of the network. In this paper, we present an evaluation which focuses on the whole network and considers metrics which are essential for evaluating *Opportunistic Networks*. Therefore, we show that our network can successfully deliver user data over multiple hops and that conversations between users can be kept maintained.

3.3.3 Paper Summary

The research on *Opportunistic Networks* is mostly theoretical with many researchers exploring possibilities and evaluating those in simulators [39]. While simulators are sufficient for initial evaluation of *Opportunistic Networks*, a network of real nodes in a realistic scenario can give more insight to the performance of the tested network. With our Android-based *Opportunistic Network opptain*, we have a network on real mobile devices that we have to evaluate outside the simulator to test thoroughly. Also, results from real-life *Opportunistic Networks* are seldom used to improve the simulators' parameters to reflect more accurate real-life networks. With an evaluation of our *opptain* application, we can both get results for our application and use these results to improve simulators. For this paper, we evaluate *opptain* and the user applications and present the test framework *oggregator* to test *opptain* with. The goal of the paper is to show the capability of *opptain* to successfully transmit and deliver data in multiple scenarios. The contributions of the paper are the development of a methodology for evaluating Android-based *Opportunistic Networks*, the field tests with a testbed of 26 devices that were used by real people and the presentation and discussion of the findings of the experiment.

We present the methodology for evaluating *Opportunistic Networks* in general and specifically Android-based *Opportunistic Networks*, consisting of the message states, the metrics and the experiment setup. Each message in the network can either be at the sending, one relaying or the destination node. The regarding states for these messages are *'generated'*, *'received'*, *'delivered'*, and *'reacted'*. Simulators and the test framework *oggregator* track these message states, as they are important for the calculation of the metrics. The used metrics are *delivery ratio*, *end-to-end delay*, *overhead*, and *hop count*. We use the two additional metrics *Client Time* and *Hotspot Time*, which are specific to Android-based *Opportunistic Networks* and important for *opptain*. The goal of the paper is to show how Android-based *Opportunistic Networks* perform in typical office scenarios that can be tested in a small environment, in this case with 26 Android devices. For the first tests, different *Time-To-Lives* (150 and 300 minutes) and routing protocols (*Epidemic* and *PRoPHET*) are the parameters that are varied

and a fixed response probability of 70 percent is used. For the field tests, the devices are spread in the university building and roughly four *communication islands* are created.

We evaluate the performance of our *opptain* network and present the results divided by metrics. The results show an average delay for all messages of about 23 minutes, an average delivery ratio of 60.85 percent, and an average hop count of about two hops. Regarding overhead ratio, for every message that is delivered, 5.15 messages are received. Regarding the metrics *Client Time Hotspot Time*, our results show that a device stays in hotspot state for about 53 seconds per cycle and in client state for 43 seconds. Also, the overall message count and the messages' states are evaluated and presented. On average, 663.38 messages are generated from scratch over a test duration of five hours and about 545 additional messages are sent as a reaction to a delivered message. 720 messages of all generated and reaction messages are delivered and 3637.62 messages are received on average. Reaction messages have an overhead of 3.11 messages, a delivery ratio of 79.6 percent and a delay of about 11 minutes. We can see that reaction messages have a lower overhead and a higher delivery ratio because answers to messages that once reached its destination have a higher chance to get back to the origin. With these numbers, we can identify possible use cases of applications for Android-based *Opportunistic Networks* in an office environment.

The test runs with a *Time-To-Live* of 300 minutes did not have a significantly higher overhead ratio than the 150 minutes *Time-To-Live* test runs. The test runs with *PRoPHET* did not have a superior ratio between overhead ratio and delivery ratio. Because of the lower *Time-To-Live*, the 150 minute test runs have a lower delay which can be explained by exceeded *Time-To-Lives* and therefore non-delivered messages. An interesting discovery is that the 150 minutes TTL *Epidemic* test runs have similar results as the 300 minutes *Time-To-Live PRoPHET* runs regarding *overhead*, *delivery ratio*, and *end-to-end delay*. The results indicate that both scenarios are suitable for use cases in which messages are exchanged and reacted to.

We can see that our *Opportunistic Network* can deliver messages successfully and reactions have a higher chance to be delivered. We can use the *Opportunistic Network* for local communication for use cases that are not in need of small delays between sender and destination. The most important point is a high delivery ratio which can also be achieved with a high overhead ratio. All scenarios rely on social interaction between the smartphone users. The parameters used for the office scenario can now be used in a simulator to research other scenarios like catastrophe or censorship situations or a village scenario. Further testing in larger scenarios, independent of real-life *Opportunistic Networks* or in the simulator, are plans for future work. All parameter studies that did not fit the scope of this paper can be done in additional field tests.

3.3.4 Personal Contribution

This paper was accepted at the 6th Workshop on Large Scale Distributed Virtual Environments co-located with the International Conference on Parallel and Distributed Computing (Euro-Par). 194 papers were submitted and 57 papers were selected for presentation and publishing, which results in an acceptance rate of 29.4%.

The author of this thesis, Andre Ippisch, contributed the idea and the conception of this paper. Philipp Brühn conducted the implementation and the experiments. The evaluation and the analysis of the evaluation results have been conducted by Andre Ippisch and Philipp Brühn. The discussion, writing and review of the paper are a joint work of Andre Ippisch, Philipp Brühn and Kalman Graffi.

Chapter 4

Improving One-Hop and Two-Hop Communication Options

In this chapter, we address the challenges regarding topology improvements for the *communication islands* in our Android-based *Opportunistic Networks*. In the last chapter, we presented the solution for our main challenge, our *opptain* application that can successfully connect Android devices; automatically and without user interaction. However, the *Hotspotting* that is used to accomplish these connections has the disadvantage that devices that are in hotspot mode cannot scan for other devices and clients can only be connected to one hotspot at a time. Also, for the first two versions of *opptain*, the devices only establish point-to-point connections and transmit data only to the connected device.

In Section 4.1, we present a third version of *opptain* in which we modify the behavior of network nodes which are connected in *communication islands* with more than two devices. We introduce an improvement of the usage of the star topology that is created if more than one device is connected to the hotspot device. In this version of the application, all devices that are connected to the hotspot device exchange meta information and can discuss the next steps for their own data transmission.

Still, devices that run this version of the application cannot scan for other devices while they are in hotspot mode. We are in need of another network interface to exchange meta information that helps to connect to other devices or indicates whether to stop the hotspot mode of a device as it might be temporarily unnecessary. In Section 4.2, we explore the possibility of smartwatches as the source of other network interfaces. The contribution of [7] is a framework to manipulate network interface of connected smartwatches and retrieve data from their sensors.

In [40], the usage of *Bluetooth* as a second signal way is explored, and we will look further into this topic as future work.

After we have explored the possibilities to improve the one-hop and two-hop communication in our *communication islands*, we next tackle the multi-hop communication in *Opportunistic Networks*. In the next chapter, we present three replication schemes to improve routing in the network.

4.1 Contact Matching and Connection Scheduling

This section summarizes the contributions of our paper [6].

Andre Ippisch, Jannik Leßenich and Kalman Graffi.
"Contact Matching and Connection Scheduling in Android-based Opportunistic Networks".
Technical Report: TR-2018-001. Technology of Social Networks Group, Heinrich Heine University, Düsseldorf, Germany. 2018.

We present the contribution of our paper in Section 4.1.1 and the importance and impact on this thesis in Section 4.1.2. We summarize the paper in Section 4.1.3 and declare the personal contribution in Section 4.1.4. The verbatim copy of our paper [6] can be found on page 0.

4.1.1 Contribution

The contribution of this paper is the enhancement of message distribution in *opptain* by preventing network fragmentation. In this paper, we introduce an advanced awareness of connection opportunities, especially for *Opportunistic Network*-specific *communication islands*. This results in an improved connection process of the Android devices and a balanced client to hotspot ratio which increases throughput in these islands. We achieve this by adding network information to the identification of network nodes and by implementing a more intelligent hotspot switching mechanism. We provide an evaluation that shows that we prevent network fragmentation in *communication islands*, and therefore stabilize the network.

4.1.2 Importance and Impact on Thesis

Because of the limitations of network interfaces on Android devices, one of the main challenges for this dissertation was to find a way to connect Android devices to each other, automatically, without user interaction, and without operating system modifications like with *rooted* devices. The only way to do this is the *Hotspotting* mechanism. In this, the feature of creating tethering hotspots is used to open access points on devices and let other devices connect to it as clients. This mechanism might be a disadvantage regarding node discovery, but as long as we have to use it, we consider all improvements regarding this mechanism as a success for our network.

Many aspects of our network are factors for the performance of our network. These aspects do not only include routing protocols including forward strategies and drop policies but also how we connect devices properly. So, all improvements regarding the connection process, transmission, throughput, and the stabilization of the network are of great importance to our Android-based *Opportunistic Network*.

4.1.3 Paper Summary

In this paper, we want to improve both the one-hop and two-hop communication in our *opptain* network by introducing the possibility to find the most suitable next connection partner for a node in the near proximity, and the exchange of information about the current neighborhood of a hotspot.

The communication islands in *opptain* occur due to the *Hotspotting* principle: Devices scan for *opptain* hotspots in their surroundings and automatically connect to one if present. If no *opptain* hotspot is present, the node itself opens a hotspot so that other devices can connect to it. This serves as a workaround for the missing ad-hoc technology on current smartphones. In this paper, we want to face the problems that the technical restrictions of Android bring. *Hotspotting* devices cannot scan their surroundings while they are in hotspot mode, and client nodes cannot connect to another hotspot without disconnecting from the current one. Because of the fluctuation of connections, it is possible that nodes miss connection partners that are important for an efficient routing of messages. The main challenge of this paper, however, is to find a pattern that ensures that every reachable node eventually finds a communication partner by reducing network fragmentation.

To avoid network fragmentation, in a communication island with many nodes it is favorable that there are as few hotspots as possible which nevertheless cover all nodes. The fragmentation of the network is divided into four subproblems: Outcasting, Clique Islands, Fake Dead Ends, and Synced Phases. Outcasting describes the situation of a node not being able to connect to the surrounding nodes, because all of them are clients for hotspots that are not in its surroundings. Clique Islands characterizes that nodes tend to connect to the same nodes over and over again, preventing message flow in the network. Fake Dead Ends are two hotspots at the edge of two node groups that stay hotspots for so long that messages cannot be exchanged between them. Synced Phases means that two nodes are synced in regards to the *Hotspotting* principle and cannot connect to each other as they always look for hotspots simultaneously.

With *Infrastructure Mode Wi-Fi* not only one but multiple nodes can connect to an *opptain* hotspot as a client. These nodes form a star-like structure in which messages can not only be delivered to the next hop but also to two-hop neighbors via the hotspot. However, this communication via the hotspot needs to be improved. In earlier versions of *opptain*, only two devices, the hotspot and a client, are communicating directly with each other. Clients have no knowledge regarding any other clients of this hotspot. If by chance a client forwards a message to the hotspot that is addressed to another of its clients, the hotspot can, in turn, forward the message. But, depending on the routing protocol, this forwarding might not happen. If the first client would know about its hotspot's connection to the second client, he would definitely forward the message.

As a contribution in this paper, we propose solutions and implementations for these problems. First, we apply a connection and message exchange scheme that will exchange neighbor lists and routing-related information. With the neighbor lists, every node has knowledge about the current neighborhood inside the communication island. Nodes can use the list and the connected two-hop neighborhood to deliver messages to their recipient without connecting to them directly. Also, we do not scan for available hotspots only once but offer the possibility to scan another time. We include flags into the *Service Set Identifier (SSID)* of a hotspot to distribute information about the node before other nodes actually connect to it. These flags include information like whether the hotspot is moving or the number of devices the hotspot has messages for. This way, the nodes can make better decisions on which hotspot to connect to. In addition to the *SSID* flags, information about the neighborhood and the last connections are considered in the decision process to find the most suitable hotspot.

For large size messages, one-hop communication is way more efficient than the two-hop communication via a hotspot. In such a case two nodes that are both connected to a hotspot can schedule an own direct connection and disconnect from the hotspot to do so. For this to work we revise the behavior clients show once their message exchange with the hotspot is finished. Whereas in the earlier version of *opptain* the connection between hotspot and client was just dissolved after the successful transmissions and the connection process started anew, the connection is now held up and can be used for further communication with the other clients. If a hotspot device is removed from the network for whichever reason, all previously connected devices will start scanning for a new hotspot. To avoid the situation of them all becoming new hotspots because they will all start scanning at the same time and might not find any active hotspot, we randomize the delay until the devices start scanning or become hotspots, which diminishes the Synced Phases problem. The other subproblems of fragmentation of the network are solved in this paper as well.

Since many of our extensions to the *opptain* application cannot be evaluated precisely with real devices that have non-deterministic behavior, we concentrate our evaluation on the results inside one communication island. We set up a field test in which all devices can connect to each other and we can expect a delivery ratio of 100%. In this environment, we achieved the delivery ratio of 100% and measured the average time the messages need to reach their destination and experienced good results. Without the improvements, the average delay was 52.4 s, with the improvements it was reduced to 38.3 s.

Therefore, in this paper, we improved the connection process by encouraging the communication with other clients inside the existing star topology and by allowing the nodes to look for more useful connection partners before connecting to a hotspot. This is achieved by exchanging meta information that inform all nodes of a star topology about the connected nodes and therefore about the connected two-hop neighborhood, as well as about the messages that should either be transferred via hotspot or in a separate direct connection that has to be scheduled.

4.1.4 Personal Contribution

The author of this dissertation, Andre Ippisch, had the idea for this paper and did the joint conception of the research. The implementation and experiments of our solution were conducted by Jannik Leßenich, as well as the joint conception of the research. Andre Ippisch and Jannik Leßenich planned the methodology, analyzed the evaluation of the experiments, and wrote and discussed the paper. Kalman Graffi contributed to the methodology of the research as well as to the revision of the paper.

4.2 An Android Wear OS Framework for Sensor Data and Network Interfaces

This section summarizes the contributions of our paper [7].

Bashkim Berzati, Andre Ippisch and Kalman Graffi.
"An Android Wear OS Framework for Sensor Data and Network Interfaces".
In: *Proceedings of the IEEE International Conference on Local Computer Networks (LCN).*
2018. Acceptance Rate: 29.8%.

We present the contribution of our paper in Section 4.2.1 and the importance and impact on this thesis in Section 4.2.2. We summarize the paper in Section 4.2.3 and declare the personal contribution in Section 4.2.4. The verbatim copy of our paper [7] can be found on page 0.

4.2.1 Contribution

In this paper, we tackle the problem that smartphones cannot scan for other devices while they are in hotspot mode. We can use the network interfaces of the wearable devices to provide the smartphone with meta information about the network. Therefore, we present a framework for Android to access and retrieve this information from the wearable devices. This data can be stored and used on all Android handheld devices. The framework is presented as a library called *ansWEARs* that can be integrated into present and future Android projects. With the library, we can not only access and manipulate network interfaces, but also retrieve sensor data from the wearable device. We analyze the library to test the reliability of the framework and present test results to show that the functions of the library can be accessed even under a high load of network traffic.

4.2.2 Importance and Impact on Thesis

The challenges of the network application *opptain* motivated the development of this framework. The smartphone nodes of the *opptain* network can either be in hotspot or client mode. While being in hotspot mode it is not possible for a smartphone by itself to scan for other

nodes. The Wear framework provides this possibility. The smartwatch can use its own *Wi-Fi* connection to receive data from other nodes and transmit it to the phone. Ultrasound was one of the possible second signal ways; the smartwatch, being capable of sending and receiving ultrasound, could take charge and forward the meta information to the phone. Also, the sensors of the smartwatch could be used, for example, for decisions based on movement or location. While the smartphone could perform both with ultrasound actions and retrieving data from movement sensors or GPS sensors, we outsource these energy consuming actions to the smartwatch and gain better results with a device that is not shielded in the user's pocket.

4.2.3 Paper Summary

To enhance the connection possibilities in our network, in this paper, we explore the possibilities and limits of the communication between smartwatches and smartphones with the aim of being able to gather and provide information about the network that is invisible to the smartphone. While we present a stand-alone Android library to manage this communication, the underlying motivation was to improve the one-hop communication of our *opptain* network. Wearables like smartwatches are gaining more interest, yet smartwatches have an abundance of sensors and network interfaces that mostly remain unused. With the background of *opptain* using the *Hotspotting* principle to connect devices, nodes are not able to scan for other nodes while being in hotspot mode, so it is favorable to have more possibilities of communication with other nearby devices. One possibility is to use the network interfaces of a smartwatch. While the *Wi-Fi* interface of the smartphone is busy with being hotspot or client, the *Wi-Fi* interface of the smartwatch can undertake further scans. We want to connect smartwatch and smartphone in a way that allows us to use the data provided by the smartwatch as well as manipulate its interfaces. To research this method and generally the capabilities of smartwatches, in this paper we create a framework in the form of a library for usage in Android projects.

To implement the framework that allows us to gather information from the smartwatch and control the interfaces, we first look into the way smartwatches are connected to a smartphone. In the Android Wear OS network, devices are nodes that are linked by *Wi-Fi* or *Bluetooth*. Multiple wearable devices can be connected to one smartphone. Android Wear OS comes with a collection of *APIs* which can be used to connect smartwatches and smartphones. We bundle these *APIs* so that smartwatch data can be transferred in a reliable and fast way and be used on the smartphone.
To make the framework as useful as possible, we declare the following demands: The data should consist of network and sensor data but also touchscreen events, whereas the framework

consists of one library and two applications, one for the wearable and one for the smartphone. We implemented the two applications, the first being the smartphone application that sends requests to the application running on the smartwatch. The wearable application collects data based on a request and sends them to the smartphone as an answer. This data can include the current network status, available and bound *Bluetooth* nodes, recorded audio samples, sensor data, and touch events. We provide the functionality as a library that can be included in all Android projects that want to exchange data with connected smartwatches.

We present an evaluation of our framework and use multiple test devices for experiments regarding the network performance, the CPU and memory usage, and the battery life. Results of the network performance tests show that large data items should be used for the communication between smartwatch and smartphone to reduce the overhead and increase the bandwidth. While the CPU tests confirm a stable CPU load on both devices, the memory tests show that it is essential to find a balance between transmission rate and package size to prevent the memory of the smartwatch from filling up before the data is received by the smartphone. The battery tests show that the *Bluetooth* negotiation between wearable and phone is the main cause for energy loss but likewise the constant gathering of sensor information drains the battery too much.

For this paper, we created a framework which helps developers and users to request and receive data from wearables to improve their applications. The conclusion of evaluating the framework is that the connection between wearable and phone must be used efficiently to preserve energy and to increase bandwidth. In the end, we predict that wearables will be an essential part of the future mobile communication. We can use the library for our *opptain* network to let nodes scan for tethering hotspots while being in hotspot mode.

4.2.4 Personal Contribution

The paper was accepted at the 3rd IEEE Workshop on Networks of Sensors, Wearable, and Mobile Devices (NSWMD 2018) which was part of the IEEE International Conference on Local Computer Networks (LCN 2018). 151 full paper submissions have been submitted to the conference from which 45 received a minimum of three reviews and have been accepted which results in an acceptance rate of 29.8%.

The author of this dissertation, Andre Ippisch, had the idea for this research and the paper and contributed the joint conception of the research. The implementation and the experiments

of the solution were conducted by Bashkim Berzati, as well as the joint conception of the research. Andre Ippisch and Bashkim Berzati planned the methodology, analyzed the evaluation of the experiments, and wrote and discussed the paper. Kalman Graffi contributed to the methodology of the research as well as to the revision of the paper.

Chapter 5

Optimal Message Replication Control

This chapter addresses the challenge of optimal message replication control in *Opportunistic Networks*.

Traditional routing protocols have the objective to minimize a certain metric, for example the *hop count*. Since in *Opportunistic Networks* the delivery of messages is not guaranteed, the objective is to maximize the probability of message delivery and to keep a low message delay [41]. Examples for traditional *routing protocols* are *DSDV (Destination Sequenced Distance Vector)* and *OLSR (Optimized Link-State Routing)* [42] in which entire routes are planned at the source node beforehand [41]. In contrast, traditional *routing protocols* in which routes are created on demand and not beforehand are *AODV (Ad hoc On-Demand Distance Vector Routing)* or *DSR (Dynamic Source Routing)* [42]. In *Opportunistic Networks*, we cannot use such traditional *routing protocols* because those assume a connected graph and require knowledge about the whole network. These connected graph structures are not guaranteed in *Opportunistic Networks* as intermittent node contacts are frequent and not an exception, and an end-to-end path between two nodes does not necessarily exist. Routing in *Opportunistic Networks* is therefore a challenge that is focused on by many researchers. Since there is no connected graph in an *Opportunistic Network*, *routing protocols* for these networks have in common that they use the *Store-Carry-Forward* principle. As nodes connect to each other, they forward or replicate the stored message, with the intention that this message or a copy of this message is carried around and reaches its destination over time.

Routing protocols in *Opportunistic Networks* can be categorized into flooding-based or utility-based protocols; however, they always use *Store-Carry-Forward* to transmit messages through the network to the destination [43]. Flooding-based protocols like *Epidemic* are the simplest

ones; they replicate a message to every connected node that has not yet received this message. Utility-based protocols like *PRoPHET* are more sophisticated and replicate messages based on a utility function that assigns each message a suitability value for the given context which can be compared for replication decisions. Because there are only a few implementations of real-world *Opportunistic Networks*, it is not clear which kind of protocol should be used by a network in which situation. For catastrophic situations, it seems reasonable to use flooding-based *routing protocols*, because they do not rely on an overview of the network. Utility-based protocols, on the other hand, make their replication decisions based on information that might not be available shortly after the establishment of the network in case that a catastrophe just occurred. However, flooding-based routing protocols use the device's resources like bandwidth and storage extensively and the high replication count implies a high overhead for the network and impacts other metrics like *delivery ratio* or *end-to-end delay* for the worse. To limit the replication for the flooding-based *Epidemic* routing protocol, multiple extensions have been created. Routing protocols like *Spray & Wait* limit the number of copies that can be present in the network and therefor require the number of copies that may still be created to be stored in the header of the message.

It is our goal to improve replication by keeping the high *delivery ratio* of the flooding-based *routing protocol Epidemic* but reducing the *overhead*, *end-to-end delay*, and resource consumption in the network. In contrast to the previously described approaches, we aim for an extension of *Epidemic* for which we only use the local view of the node and do not have to add specific information about the network to the header of a message.

In the next sections, we offer solutions for the challenge of optimal message replication, as source and relay nodes need to have some kind of replication control for messages when searching for the destination. The three papers [11, 12, 13] presented in this chapter build on one another and differ in essential aspects. They address the question on when to stop flooding a message, because the delivery probability is not high enough to justify the usage of the scarce bandwidth that could instead be given to other messages. We want to find a trade-off between *delivery ratio* and resource consumption, not only regarding bandwidth but also storage. In the following, we give an overview of the commonalities and differences of the papers.

With *Most Of Storage and Transmission – with Replication Probability Threshold (MOST-RPT)* [11], we create a *forward strategy* which defines a delivery probability formula that is based on the *hop count* and *replication count* of the messages, makes for an optimal number of message copies in the network and hence improves the consumption of system resources. Only if the calculated delivery probability is larger than a dynamic threshold (thus the name of the

paper "Dynamic Replication Control Strategy"), the message is forwarded. This threshold is set based on the *Inter-Contact Time* of the node and the *Time-To-Live* of the message. The delivery probability of our *forward strategy* only decides if a message should be forwarded but not in which order; for the order of the forwarding queue, we use *First-In-First-Out (FIFO)*. If the buffer is full, the *drop policy* of our model evicts all messages whose probability is smaller than the dynamic threshold. If there are no messages meeting this condition, *First-In-First-Out (FIFO)* is used to remove the oldest message in the queue. For our evaluation we use *Epidemic* as the reference protocol that *MOST-RPT* is compared to.

With *Replication Probability-based Routing Scheme (RPRS)* [12], we extend our replication scheme *MOST-RPT*. With this approach, we can further optimize the usage of system resources and improve the *delivery ratio* while lowering the *overhead* in the network. While *MOST-RPT*, compared to *Epidemic*, only improved the *drop policy* and used the *First-In-First-Out (FIFO)* approach for the scheduling of messages, *RPRS* provides a new *queueing policy*. The forwarding queue is ordered by delivery probability, with the message with the highest probability being forwarded first. The *drop policy* of *RPRS* calculates a weight based on the storage costs for the buffered message, which are the sum of the buffer times at source and relay nodes, and transmission costs. The message with the highest weight is dropped first. For our evaluation, we use *Epidemic* and *Spray & Wait* as the reference protocols to compare *RPRS* to; *Epidemic* for our performance metrics *delivery ratio*, *overhead* and *end-to-end delay*, and *Spray & Wait* for the amount of delivered messages and the buffer delay.

With *Optimal Replication Based on Optimal Path Hops (ORBOPH)* [13], we derive an optimal hop and replication count which are used for the computation of an optimal threshold for the delivery probability of *MOST-RPT*. Hence, in contrast to *MOST-RPT* and *RPRS*, the threshold for the decision whether a message is forwarded is not dynamic. The ideal threshold optimizes the distribution of messages in the network by minimizing the amount of message copies. Reusing the policies of the two previous approaches whose combination provides good results, we use the *queueing policy* of *RPRS* and the *drop policy* of *MOST-RPT*. Thus, the message with the highest probability gets forwarded and the *drop policy* removes the message with the smallest probability first. For our evaluation, we use *Epidemic* as the reference protocol to compare *ORBOPH* to.

Additionally, for all replication models, the buffer is not only emptied by the mentioned *drop policies* but also when messages expire based on their *Time-To-Live*. Another approach, to keep expired messages in the buffer as long as there is enough space and subsequently replicate the message only directly to the destination, is presented in the future work section 6.2.

In Section 5.1, we present *MOST-RPT*, in Section 5.2, we present *RPRS*, and in Section 5.3, we present *ORBOPH*.

Importance and Impact on Thesis

In Chapter 2, we introduced our network application *opptain* which contains multiple *routing protocols*, *forward strategies*, and *drop policies* from related literature and is designed to let developers add new protocols, strategies and policies in a simple matter. Examples of those included by us, for routing these are flooding-based protocols like *Epidemic* or utility-based protocols like *PRoPHET*, for *forward strategies* and *drop policies*, *First-In-First-Out (FIFO)* and *Evict-Most-Forwarded-First (MOFO)* are two examples. In Chapter 4, we utilized the star topology of our *opptain* network to improve one-hop and two-hop communication in isolated islands. To improve multi-hop communication in *Opportunistic Networks*, we have to address *routing protocols*, *forward strategies* and *drop policies*. A variety of protocols and policies can be useful for different situations and this diversity is an advantage for the productive usage of the network as well as for experiments in a test environment. This is why all routing protocols, *forward strategies* and *drop policies* in *opptain* are interchangeable by the developer, and in the current version by the user, too. Flooding protocols like *Epidemic* are essential for our *opptain* network, especially for catastrophic situations; however, flooding protocols use the device's resources extensively. Therefore, replication schemes like *MOST-RPT*, *RPRS* and *ORBOPH* make *opptain* more resource-friendly which is essential for our project and this thesis.

The detailed importance and impact of our replication schemes can be found in the following sections, with *MOST-RPT* in Section 5.1.2, *RPRS* in Section 5.2.2, and *ORBOPH* in Section 5.3.2.

5.1 Dynamic Replication Control Strategy

This section summarizes the contributions of our paper [11]. It is the first of three papers of this thesis about replication control in *Opportunistic Networks*. Here, the base for our replication schemes is set by creating a formula for delivery probability based on the *hop count* and *replication count* of the messages. The goal is to achieve a near-optimal replication to provide a high delivery ratio and a low *overhead* in the network.

Salem Sati, Andre Ippisch and Kalman Graffi.
"Dynamic Replication Control Strategy for Opportunistic Networks".
In: *Proceedings of the International Conference on Computing, Networking and Communications (ICNC)*. 2017. Acceptance Rate: 29%.

We present the contribution of our paper in Section 5.1.1 and the detailed importance and impact on this thesis in Section 5.1.2. We summarize the paper in Section 5.1.3 and declare the personal contribution in Section 5.1.4.

5.1.1 Contribution

The contribution of this paper is a replication scheme consisting of a *forward strategy* and a *drop policy* for *Epidemic* routing, called *Most Of Storage and Transmission – with Replication Probability Threshold (MOST-RPT)*. In *Opportunistic Networks*, an end-to-end path between nodes is not guaranteed and messages are delivered via *Store-Carry-Forward*. To reduce the above-mentioned *overhead* in the network, in this paper, we look into the replication control of *Epidemic* routing, one of the flooding-based routing protocols for *Opportunistic Networks*. Our replication scheme consists of, first, a delivery probability policy which integrates the message's *hop* and *replication count*, and second, a model for the desired number of message copies in the network. To formulate our policy, we extend a mathematical model from related literature. *MOST-RPT* decides if a message should be spread and we reach a high *delivery ratio* without congesting the network. We conduct an experiment to compare our replication scheme to *Epidemic* and present numerical results that show that our dynamic replication scheme performs close to the optimal values of our mathematical model.

5.1.2 Importance and Impact on Thesis

The importance and impact of *MOST-RPT* on this thesis lies in the improvement of *Epidemic* to reduce the *overhead* in the network while keeping the *delivery ratio* high. An overview of the importance and impact of this topic on this thesis can be found in the introduction of Chapter 5.

5.1.3 Paper Summary

Opportunistic Networks are networks without guaranteed end-to-end path between nodes in which *Store-Carry-Forward* is used to route messages from source to destination. Routing in these networks can be classified into flooding-based and utility-based routing, with *Epidemic* and *PRoPHET* as representatives respectively. With the flooding-based approach of *Epidemic*, we achieve a high *delivery ratio*, but with the cost of a large *overhead*. To reduce this *overhead*, in this paper we propose a replication scheme for *Epidemic*, called *MOST-RPT*. With *MOST-RPT*, our goal is to achieve a near-optimal message replication which provides both a maximum possible message delivery ratio and an *overhead* as small as possible. The replication scheme utilizes each message's *hop* and *replication count* for its decisions and consists of a rule for the optimal number of message copies in the network.

For our solution, we use multiple individual models from related work to form our overall system model. The communication model defines a network with a finite number of nodes, the communication interfaces like *Wi-Fi* or *Bluetooth*, as well as limited buffer and energy. The mobility model is characterized by the nodes' *Inter-Contact Time* and their contact duration. The *Inter-Contact Time* is the time between node meetings and has a high impact on the delay in the network. The contact duration is the time two nodes are connected for. The routing model assumes unique identification for nodes, a single destination, and a *Time-To-Live* for each message. We choose *Epidemic* to focus on resource consumption, for solving by *Ordinary Differential Equation*, and because it is the most used protocol in middleware. To address resource constraints, mostly the limited buffer, we focus on the buffer space and the order of the messages. The main aim for our replication scheme is to maximize the global *delivery ratio* in the network. Our dynamic replication policy consists of a *forward strategy* and a *drop policy* which use a utility function based on the *hop* and *replication count*. We present the analytical model based on a *Markov Chain model* and a *Ordinary Differential Equation* that results in the definition of our *forward strategy* and a *drop policy*.

The *forward strategy* of *MOST-RPT* defines a delivery probability formula that is based on the *hop count* and *replication count* of the messages, which both characterize the messages' *overhead* in the network. Based on the delivery probability, the *forward strategy* decides whether it is likely enough that the message will reach its destination by comparing it to a threshold. If the delivery probability exceeds the threshold, the message is forwarded. If it does not exceed the threshold, it is not forwarded, but still kept in the buffer as long as the buffer is not full and the *Time-To-Live* has not run out. The threshold is calculated dynamically by taking the *Inter-Contact Time* of the node and the *Time-To-Live* of the message into account. *MOST-RPT* utilizes the same *queueing policy* as *Epidemic*: *First-In-First-Out (FIFO)*. If the buffer is full, the *drop policy* evicts those messages whose probability is smaller than the dynamic threshold. If there are no messages meeting this condition, *First-In-First-Out (FIFO)* is used here as well.

For the evaluation, we present multiple scenarios to get results regarding our performance metrics. We use the three quality metrics for evaluation that are most important to *Opportunistic Networks*: *delivery ratio*, *overhead*, and *end-to-end delay*. We use three different scenarios to reflect different traffic situations; these differ in the combination of node types (pedestrians, cars, trains), buffer size, message creation interval, message size, and *Time-To-Live*. We use *the ONE simulator* to conduct the tests and compare our scheme with five different *drop policies*. The results show that *MOST-RPT* can be a good base for future utility-based *routing protocols* to achieve a better performance in *Opportunistic Networks*. For fast traffic, *MOST-RPT* has the highest delivery ratio compared to the other policies, although only leading by 1% to the second best, which is the reference *drop policy Evict-Most-Forwarded-First (MOFO)*. The same holds for slow traffic, with one exception where it comes in second best after *Evict-Most-Forwarded-First (MOFO)*. For *overhead*, the main performance factor for comparison, *MOST-RPT* achieves the best results compared to the other policies. Only for *end-to-end delay*, *MOST-RPT* is only second-best after the policy *MaxRep* which drops messages with a high replication count first. *MaxRep*, however, has the lowest delivery ratio and a high overhead. The high *delivery ratio* as well as low *overhead* and *end-to-end delay* of *MOST-RPT* are very promising for our overall network. With *MOST-RPT*, we have a balance between the three quality metrics that is a good base for future research on the topic of replication control in *Opportunistic Networks* and motivates the next contribution which we show in Section 5.2.

5.1.4 Personal Contribution

This paper has been accepted at the International Workshop on Computing, Networking and Communications (CNC 2017) which was part of the 6th International Conference on Computing, Networking and Communication (ICNC 2017). Each submitted paper has received three technical reviews from the Technical Program Committee. The acceptance rate was 29%.

The author of this dissertation, Andre Ippisch, implemented the forwarding and drop policies in the ONE simulator, conducted a set of experiments to demonstrate the results, and contributed to the methodology and to the revision of the paper. The contributions of Salem Sati are the drafting of the methodology, solution and paper. He also contributed to the joint development of the mathematical analysis and the joint conduction of the experiments. Kalman Graffi contributed to the methodology of the research and revised the paper.

5.2 Replication Probability-based Routing Scheme

This section summarizes the contributions of our paper [12]. It is the second of three papers of this thesis about replication control in *Opportunistic Networks*. In previous work [11], we created a replication scheme using a formula for delivery probability that indicated whether the message should be forwarded. In this paper, we aim to further reduce the *overhead* in the network. We replace *MOST-RPT*'s *drop policy* by a new one based on storage and transmission costs. Additionally and in contrast to *MOST-RPT*, we define our own *queueing policy* to replace *First-In-First-Out (FIFO)*.

Salem Sati, Andre Ippisch and Kalman Graffi.
"Replication Probability-based Routing Scheme for Opportunistic Networks".
In: *Proceedings of the GI/ITG International Conference on Networked Systems (NetSys)*.
2017. Acceptance Rate: 42.9%

We present the contribution of our paper in Section 5.2.1 and the detailed importance and impact on this thesis in Section 5.2.2. We summarize the paper in Section 5.2.3 and declare the personal contribution in Section 5.2.4.

5.2.1 Contribution

In this paper, to further reduce the *overhead* in *Opportunistic Networks*, we present a second attempt for a replication scheme for *Opportunistic Networks*, called *Replication Probability-based Routing Scheme (RPRS)*. We propose a *forward strategy* and a *drop policy* based on *Epidemic* routing and the replication scheme *MOST-RPT* from our previous paper. This scheme is based on controlled message replication and aims to keep the *delivery ratio* high while reducing the *overhead* in the network. In contrast to *MOST-RPT*, which uses *First-In-First-Out (FIFO)*, *RPRS* proposes a *queueing policy* that is based on the delivery probability. The proposed *drop policy* uses a utility function which includes *replication* and *hop count*, and the buffer time of the message, which is an estimation of the *end-to-end delay*. In our experiments, we analyze the performance of *RPRS* by comparing it to *Epidemic* and *Spray & Wait* with multiple *forward strategies* and *drop policies*. We show that *RPRS* produces better performance values in terms of *delivery ratio*, *end-to-end delay*, buffer delay, and *overhead* than all reference protocols.

5.2.2 Importance and Impact on Thesis

The importance and impact of *RPRS* on this thesis lies in the improvement of *Epidemic* to reduce the *overhead* in the network while keeping the *delivery ratio* high. An overview of the importance and impact of this topic on this thesis can be found in the introduction of Chapter 5.

5.2.3 Paper Summary

Opportunistic Networks are delay tolerant networks without a guaranteed end-to-end path between nodes in which *Store-Carry-Forward* is used to route messages from source to destination. With a flooding-based routing approach like *Epidemic*, we achieve a high *delivery ratio* with the cost of a high *overhead*. To lower the *overhead* and keep a high *delivery ratio*, in this paper, we propose a new replication scheme for *Opportunistic Networks*, which consists of a replication controlled *forward strategy* and a *drop policy*; the replication scheme is called *Replication Probability-based Routing Scheme (RPRS)*. This replication scheme is oriented towards *MOST-RPT* by reusing the delivery probability formula but stands for itself by introducing its own *forward strategy* and *drop policy*. For our solution, we first describe a system model which is based on a *Markov Chain model* with an *Ordinary Differential Equation*. Next, we define the *RPRS* replication control strategy. As in *MOST-RPT*, the *forward strategy* uses the delivery probability for decisions on forwarding or replicating messages, and as a result of this, an optimal replication count for a message in the network is kept at each node. This increases *delivery ratio*, and decreases *overhead* and *end-to-end delay*. Instead of applying *First-In-First-Out (FIFO)*, *RPRS* introduces its own *queueing policy* and arranges the forwarding queue based on delivery probability. The message with the highest probability is forwarded first. *RPRS* introduces a different *drop policy* than *MOST-RPT*. *RPRS* calculates a weight for each message and removes the messages with the highest weight from the buffer. For this weight calculation the buffer times of the source and the relay nodes are used, which characterize the storage and transmission costs in the network. Both replication and drop criteria, are calculated based on this local message information. The main idea of *RPRS* is that the best *delivery ratio* is achieved when the forwarding is adapted to the dropping and the buffer is optimally filled. We also explain the difference to existing controlled quota and probability *routing protocols*: *RPRS* is an heuristic scheme which uses only local message information, calculates replication criteria dynamically, uses the buffer time for its drop policy and uses no quota.

For the evaluation of our scheme, our performance metrics are *delivery ratio*, *overhead*, *end-to-end delay*, average buffer delay, and the amount of delivered messages. The performance of *RPRS* is compared to two flooding-based routing protocols, *Epidemic* and *Spray & Wait*. We use different *forward strategies* and *drop policies* for comparison. We use a Random Waypoint model as the mobility model to allow a better comparison to other *routing protocols* in literature. To reflect multiple traffic situations, we conducted three different scenarios with different traffic parameters and *Time-To-Live* values for experiments with *Epidemic*, each scenario being different in forwarding and drop policies. For all three scenarios, *RPRS* shows either the best or an equally good performance for each single performance metric. Regarding the combination of the metrics, *RPRS* performs better in all scenarios. Also, we compare the performance of *RPRS* to *Binary Spray & Wait*. For this comparison, the amount of delivered messages and the buffer delay are used as performance metrics. For all scenarios, *RPRS* shows more delivered messages and less buffer delay. By using the *replication count*, *hop count*, and buffer time, the issues of improving replication in *Opportunistic Networks* are addressed. *RPRS* considers the trade-off between replication and resource consumption in *Opportunistic Networks*. The performance evaluation shows that *RPRS* performs better regarding the main performance metrics in comparison to *Epidemic* and *Spray & Wait*.

5.2.4 Personal Contribution

The International Conference on Networked Systems (NetSys 2017) accepted 18 of 42 submissions which results in an acceptance rate of 42.9%.

Salem Sati did the conception of the research, drafted the paper, developed the mathematical analysis part, and did the joint conduction of the experiments and their evaluation. The author of this thesis, Andre Ippisch, implemented the necessary functionality, conducted a set of experiments and evaluated these, and revised the methodology of the paper. Kalman Graffi revised the methodology and the paper.

5.3 Optimal Replication Based on Optimal Path Hops

This section summarizes the contributions of our paper [13]. It is the last of three papers of this thesis about replication control in *Opportunistic Networks*. In previous work [11, 12], we created two replication schemes based on a formula defined in *MOST-RPT* and showed that both replication schemes reduce the *overhead* in our network while keeping the *delivery ratio* high. However, our goal is to reduce the *overhead* even more by considering the current network structure. We derive an optimal *hop* and *replication count* and use them for our replication scheme to optimize the distribution of messages by minimizing the amount of message copies. Furthermore, we combine the *queueing policy* and the *drop policy* of *RPRS* and *MOST-RPT* respectively.

Andre Ippisch, Salem Sati and Kalman Graffi.
"Optimal Replication Based on Optimal Path Hops for Opportunistic Networks".
In: *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA)*. 2018. Acceptance Rate: 28%

We present the contribution of our paper in Section 5.3.1 and the detailed importance and impact on this thesis in Section 5.3.2. We summarize the paper in Section 5.3.3 and declare the personal contribution in Section 5.3.4.

5.3.1 Contribution

The contribution of this paper is a replication scheme for *Opportunistic Networks*, called *Optimal Replication Based on Optimal Path Hops (ORBOPH)*.
In this paper, to further reduce the *overhead* in a network, we use a different approach to extend our previous replication schemes. This new approach uses flooding-based routing protocols and exploits the mobility and topology in the network. The improved scheme, *ORBOPH*, is based on a mathematical model which combines the delivery probability with a limitation on the maximum number of hops. We define constraints on the amount of replications and hops for each message to create a replication policy which increases the efficiency of resource allocation and decreases the *overhead* in the network. The results are an improved forwarding strategy and multiple drop policies for our replication scheme. We perform experiments to compare our policies to those of current literature and present evaluation results which show that *ORBOPH* performs better than *Epidemic*.

5.3.2 Importance and Impact on Thesis

The importance and impact of *ORBOPH* on this thesis lies in the improvement of *Epidemic* to reduce the *overhead* in the network while keeping the *delivery ratio* high. An overview of the importance and impact of this topic on this thesis can be found in the introduction of Chapter 5.

5.3.3 Paper Summary

In this paper, we approach the challenge of high mobility and unreliable communication in *Opportunistic Networks*. In *Opportunistic Networks*, it is our goal to have a high *delivery ratio*, and a low *overhead* and *end-to-end delay*. One approach to control these metrics is to limit the number of hops in the network path of a message. We aim for a routing scheme that defines a favorably optimal limit for the message's hop count. A solution for routing in the network are routing schemes with an optimal number of hops to get messages to their goal. Due to their properties, in *Opportunistic Networks* this optimality cannot be achieved completely; however, mobility and topology can be exploited to increase the routing performance.

We present *ORBOPH*, a replication scheme for *Opportunistic Networks* based on optimal path hops in the network. This replication scheme is an extension of *MOST-RPT* and *RPRS*. Our goal is to find the optimal number of replications of a message to increase the efficiency of the resource allocation and reduce the *overhead* in the network. With an estimated optimal number of replications and hops, we calculate a threshold for the delivery probability that was defined in *MOST-RPT*. In contrast to *MOST-RPT* and *RPRS*, *ORBOPH* uses no dynamic threshold. The used *hop count* and *replication count* are estimated by means of the current network structure to improve the distribution of messages in the network by minimizing the amount of message copies. With *ORBOPH* we combine the *queueing policy* and the drop policy of *RPRS* and *MOST-RPT* respectively: the messages with the highest delivery probability are forwarded first, whereas those with the lowest probability are removed first.

For the evaluation, we first present the used performance metrics, which are *delivery ratio*, *overhead* and *end-to-end delay*. We compare *ORBOPH* with *Epidemic* and different replication and drop policies. We also use multiple scenarios for testing, each with different *Time-To-Live* values. The first scenarios use different options for their buffer management and forward queue to compare with *ORBOPH*, and the last scenario compares different *hop count* values of *ORBOPH* to confirm the estimated *hop count*. We perform all tests with *the ONE simulator*.

For all scenarios and all performance metrics, *ORBOPH* achieves the best values, either directly or in the overall picture. Also, the estimated hop count value is confirmed by the last scenario test. In this paper, we present the *ORBOPH* model which is constructed with related analytical studies that use omni-directional infection. The results of our experiments show that with improved node infection, routing schemes based on *Epidemic* can perform better. For future work, we want to compare our three different replication schemes to each other.

5.3.4 Personal Contribution

The 32nd IEEE International Conference on Advanced Information Networking and Applications (AINA 2018) accepted 157 submissions with an acceptance rate of about 28%.

The author of this thesis, Andre Ippisch, initialized the focus on human carried devices and the mobility in these networks, and Salem Sati initialized the mathematical analysis part of the paper. The joint drafting of the methodology, solution, and paper, as well as the further development of the mathematical analysis part of the paper and the conduction of the experiments and their evaluation was done by Andre Ippisch and Salem Sati. Andre Ippisch additionally implemented the policies. Kalman Graffi contributed to the methodology of the research as well as to the revision of the paper.

Chapter 6

Conclusion & Future Work

In this thesis, we met the challenges of creating an *Opportunistic Network* for Android smartphones and improving routing in *Opportunistic Networks* in general. In the first chapter of this thesis, we defined our problem statements, derived research questions and challenges from these and answered the questions and solved the challenges throughout this dissertation.

Over the years, smartphones have become more powerful and omnipresent in our daily lives. With increasingly powerful processors and graphics cards, large storage, longer-lasting batteries, and several options included for connecting to the Internet, smartphones offer a broad platform for the high number of apps that are available and offer us a variety of functionalities. However, without an active connection to the Internet, most of the current apps are rather useless. Offering *Opportunistic Networks* for smartphones provides an alternative to the Internet. There are many scenarios in which the Internet might be unavailable as the network to route messages over. In some areas of the world the Internet is actually not available to begin with, and even if it is, the usage might be restricted due to censorship. In catastrophic situations, the Internet's infrastructure could be destroyed, making any connection impossible. The possible unavailability is not the only reason for offering an alternative to the Internet; using it is also not always desirable. Every message leaves a trace on the Internet that, depending on the content or the user, can be unwanted; for example, activists might want to share their information in a more private way. Sending a message across the Internet when you are standing right next to each other seems unnecessary and can be replaced by local communication. This local communication might not only be faster than the routing through the Internet, but it also requires no further infrastructure other than the smartphone and hence implies fewer costs. Hence, for smartphones, *Opportunistic Networks* are a useful addition to the Internet. Looking at this in another way, we can say that it is relatively simple to motivate smartphones

as nodes in *Opportunistic Networks*. It is far easier to get people to use another connection technique and another app on the smartphone they already own anyway than to get them to buy an extra piece of hardware to take part in the *Opportunistic Network*. Also, the observed mobility patterns of people in the formation of groups and their consequent movements to other groups suits an *Opportunistic Network* very well.

Although smartphones and *Opportunistic Networks* are a perfect match, there has not been much work on this combination and actually not even a lot of work on the use of *Opportunistic Networks* themselves in practice. To change that, several challenging issues have to be resolved in the establishment of an *Opportunistic Network* with smartphones: node discovery, connection establishment, time and buffer management, identification of nodes, encryption of messages, security threats, efficient routing, and also the distribution of our app, as the stores that are normally used are not available without the Internet. While overcoming these hurdles is supposed to result in the establishment of an *Opportunistic Network* that can actually be used, we also aim at creating a test environment which can be utilized by us and other researchers to evaluate protocols that have been previously tested in simulators in a real-world setting.

In the following section, we give detailed conclusions for each of our research topics defined by the problem statement and research questions. We will both describe our solutions and state their outcomes. Afterward, we present future work and give some closing remarks.

6.1 Conclusion

In this section, we conclude the work from our publications regarding the various topics in the field of *Opportunistic Networks*. We take a look at each problem statement defined before starting this dissertation as well as the problem statements that arose during the work. To structure the multitude of problem statements, we choose to organize the topics based on the classification of the research questions and the developed contributions.

The Creation of an Android-based Opportunistic Network

For the main challenge of this dissertation, we introduced *opptain*, our Android-based *Opportunistic Network*. We decided to use Android smartphones as the devices that form the

Opportunistic Network since Android is the most used operating system on smart devices. Furthermore, Android allows us to meet our demands, although we had to find a workaround that fulfills the demands for no user interaction, automatic node discovery and connection establishment since unfortunately smartphones are not able to use ad-hoc networking. With the principle of *Hotspotting*, devices can connect to each other automatically and without user interaction. With private/public keys we can both encrypt messages in the network and provide identification of devices in the form of the fingerprint of the public key. We facilitate the exchange of public keys between users via functions to show and scan a *Quick Response Code (QR code)*. We provide an *API* to connect user applications to our network application *opptain* and provide user applications, a messaging app and a filesharing app, as proof of work. We evaluated the network and user applications and showed that the principles of *Opportunistic Networking* fit Android smartphones. With this, our main challenge of creating an Android-based *Opportunistic Network* is solved and we turn to the newly emerged challenges.

In its first version, the *opptain* application which runs on off-the-shelf Android smartphones assumed synchronized clocks and a free and infinite storage. As these assumptions do not hold, we encountered two more challenges. First, we discussed the time management in *Opportunistic Networks*. If the nodes' clocks are not always set the same, they at least have to know how to handle different times. Also, the smartphone's storage must be divided correctly and the buffer of our network application must be emptied at the right moments to keep the storage from filling up. Mechanisms like *NTP* rely on the Internet and are not available in our network. Also, Android prevents non-system applications from changing the clock, so clock synchronization based on devices learning about the clocks of other devices is not an option for our network. Instead, we work with clock differences and store these differences along the path of our message in the network. For critical aspects of the network like a working *Time-To-Live* the time differences can be used to calculate the correct time specifications. Also, user applications can use this information to bring messages into the correct order. Second, we discussed the buffer management in *opptain*. Devices specifically designed for *Opportunistic Networks* have no requirements regarding the buffer, but on Android smartphones, the storage is shared among all applications. On the one hand, *opptain* should use all the available storage for its buffer, but on the other hand, it should also react to a full storage, even though that might have been filled by other applications, because a completely filled storage disables the phone. Android lacks a reliable warning system indicating that the storage is running full, thus to prevent the storage from being crammed, we have to check its current capacity periodically. Concerning the decision as to which messages to keep or remove, we have to use appropriate forward and drop policies. We included two modules in our *opptain* application to offer support for the desired time and buffer management.

In the course of this dissertation, we included a multitude of modules that make up the *opptain* framework. We implemented modules for the search for *opptain* hotspots in the near proximity and the evaluation of these based on past connections and current information, and the automatic connection to *opptain* hotspots or the opening of a new hotspot if none are available nearby. We provide a module for the creation of a node identity by generating a public/private key pair and using the public key's fingerprint as identification. Furthermore, we created modules for the encryption and signing of messages with the generated public/private key pair, and the decision about forwarding and replication of messages based on a variety of routing protocols. We provide modules for the buffer management on Android devices and for dealing with time changes on a single device and time differences between devices. Also, we provide an *API* to connect user applications and a module for the replication of *opptain* and its user applications to new participants. With these modules, the implementation of our *opptain* framework according to the design and demands of this work is finished.

Studies and Evaluation for our Android-based Opportunistic Network

Before dealing with any other issues and developing more improvements, we went on to evaluate this version of *opptain*. For a first evaluation of *opptain*, we look into the very important aspect of node density. An *Opportunistic Network* is realized by nodes that have a physical presence in a bounded area and only if two nodes are close to each other can a connection be established and messages can be exchanged. The whole spectrum of node density must be viewed. If we take Earth as an example of the area in which an *Opportunistic Network* is established, one participant per country would be a perfect example of an *Opportunistic Network* that is doomed to failure. For the other special case, a lecture hall with hundreds of students communicating by using *Opportunistic Networking* can present a problem. Modern routers are already struggling with high amounts of users in close-range, but local nodes with a rather less powerful network card/interface cannot handle that many simultaneous connections. Smart connectivity establishment as it is applied in *opptain* can use star topologies or connection control to handle a multitude of devices, but hundreds of nodes might still pose a problem. A proper balance between these two extremes probably results in better outcomes. To examine the outcome of different node densities, we conducted experiments with varying amounts of nodes in bounded areas. Furthermore, not only the amount of nodes per area is important, but also the transmission range of the devices.

Next, we proposed a study on security threats in Android-based *Opportunistic Networks*. We used literature to determine common security issues in *Opportunistic Networks* and analyzed

our *opptain* application accordingly. We proposed mitigation techniques and implementations to tackle several issues and threats. However, some security risks can be decreased but not solved completely, as is the case for *Opportunistic Networks*, for which anonymity is a blessing as well as a curse. For the Android-specific security issues, we developed solutions that are included in *opptain* as well.

To conclude our evaluations, we conducted a field experiment with our *opptain* application and presented the analysis of these tests. In contrast to evaluating results in a simulator, it is a challenge to evaluate our Android-based *Opportunistic Network*. For both, the network and its evaluation, many aspects influence their quality. As mentioned before, the node density plays a crucial role in an operative network. For our tests, a small group of people from two neighboring departments of the university participated in a field experiment to test the *opptain* network. We evaluated the results and confirmed that in scenarios with a suitable node density messages can be exchanged successfully. With this field experiment, we accomplished all the intended evaluation. Still, there are uncountable scenarios and processes that need to be tested thoroughly, which is open for future work. The results of our experiments can be used for simulators; however, not all current *Opportunistic Network* simulators support our Android-based connection principles. We extended *the ONE simulator* by these connection principles to allow further precise testing even without real Android devices.

Improving One-Hop and Two-Hop Communication Options

Having resolved the primary challenges, we turned to new questions regarding the one- and two-hop communication in our network. To improve the selection of connection partners and the insight of each node on the surrounding network, we worked on a second signal way and the meta data exchange.

To enable additional signal ways and the retrieval of useful data, we developed the framework *ansWEARs* to use smartwatches' network interfaces and sensors. One of the problems of the first challenge of this thesis was how to connect multiple smart devices with each other. Due to the lack of ad-hoc networking on smartphones, we decided on the *Hotspotting* mechanism and used it as the only way to communicate with other devices. Smartwatches can be connected to the smartphone by *Bluetooth Low Energy*. However, smartwatches have other interfaces in addition to *Bluetooth*, for example a working *Wi-Fi* chip and also sensors that can be used as an aid for our network. The *Wi-Fi* chip, for example, can be used to search for other hotspots in the surroundings while the smartphone is in hotspot mode, which provides information

about the network that the smartphone can then benefit from. *GPS* data that is useful for routing protocols can be obtained from the watch which is mostly worn outside of pockets and therefore offers more precise results than the smartphones' *GPS* receiver. The evaluation of the framework shows that the network interface can be used to meet the initial desire and that we can obtain useful sensor data. Thus we offered a way to use smartwatches as a second signal way.

To improve the connection process inside the communication islands of our *opptain* application, we revised the meta data exchange. So far, we had developed a mechanism to connect devices to each other in a simple fashion: Either there is a tethering hotspot that our device connects to or the device itself becomes a hotspot. However, while being a hotspot the device cannot scan for other devices. When relying only on *Wi-Fi*, for two devices that are both in hotspot mode it will potentially take a long time until they find each other and set up a connection. This implies that nodes might miss beneficial possible connection partners. Furthermore, if there are many devices connected to one hotspot device, the prior mechanism just disconnected the devices from each other after successful transmissions and the connection process started anew. In our paper, we improved this connection process by exchanging meta data inside the existing star topology and by allowing the nodes to look for better connection partners before they disconnect and decide on a hotspot. Of course, improvements of this kind are an ongoing process and there is still room for more enhancements.

Optimal Message Replication Control

Since we designed *opptain* to be flexible enough to try out different *routing protocols*, we were able to perform experiments with the goal to improve flooding-based routing algorithms. We presented three approaches that build on each other and that improve the routing with different *forward strategies* and *drop policies*.

First, we proposed *MOST-RPT* and, based on *hop count* and *replication count*, included a formula for the delivery probability of a message. Messages are only forwarded if the calculated probability is larger than a dynamic threshold. The messages whose delivery probability is smaller than this dynamic threshold are kept only until the storage is full and messages need to be deleted, in which case they are evicted by the *drop policy*, starting with the message with the smallest probability. Additionally, *First-In-First-Out (FIFO)* is used for the order in which the messages are forwarded and also removed if all messages have a high enough probability.

Second, we proposed *RPRS*, which also uses the delivery probability as defined for *MOST-RPT* to decide which messages to forward. Other than *MOST-RPT* using *First-In-First-Out (FIFO)*, the forwarding queue in *RPRS* is ordered by delivery probability and the message with the highest probability is forwarded first. The *drop policy* is replaced as well. Instead of removing the message with the lowest delivery probability, each message has a certain weight, calculated from storage consumption and transmission costs, and the message with the highest weight gets deleted first.

Third, we presented *ORBOPH*. We derived an optimal *hop* and *replication count* and inserted them into the delivery probability formula of *MOST-RPT* to calculate the threshold that a message's delivery probability has to exceed for it to be forwarded. This optimizes the distribution of messages in the network by minimizing the number of message copies. The forwarding order of the messages is based on the probability as well so that the one with the highest probability is forwarded first. If the buffer is full, the *drop policy* removes the message with the smallest probability first.

Each of our solutions was evaluated in *the ONE simulator* and the results show that they perform better than the reference protocols. Because in an *Opportunistic Network* there is always a correlation between resources and outcome and there is a high amount of possibilities to improve the replication control, the challenge can and will be tackled more often.

With the design, implementation and evaluation of our smartphone framework to apply *Opportunistic Networking* in a real-world setting, we accomplished our goal for this thesis. However, there are some questions and challenges that have been left open, which we will present in the next section.

6.2 Future Work

In the course of the implementation of our *opptain* application and the research on *Opportunistic Networks* in general, we came across new questions and challenges which we could not yet answer. In this section, we want to introduce these questions and challenges as future work.

With newer Android versions, an implementation of *infrastructureless Wi-Fi* called *Wi-Fi Aware* was introduced by Google; however, for now, only to Google's flagship devices. Additionally, the *APIs* for usage of this service still had unresolved issues which made it unusable.

According to the issue tracker of Google, these issues are now resolved and a working *Wi-Fi Aware* is expected in the next version of Android. For future work, it would be beneficial to adapt *opptain* to use *Wi-Fi Aware* on Android devices.

If a true ad-hoc solution is integrated to our *opptain* network, additional improvements regarding the message dissemination process inside the *communication islands* of our *Opportunistic Network* are desirable. These sub-networks can be viewed as peer-to-peer networks and then improved by using routing and monitoring approaches [44, 45, 46] and by testing them in the peer-to-peer simulator PeerfactSim.KOM [47, 39]. This should give more insight into the replication process in these *communication islands*.

While looking into possibilities to exchange data with other devices, we expanded our spectrum (literally) by considering ultrasound. *Bluetooth* as a second signal way has the disadvantage that it uses the same frequencies as *Wi-Fi*, which causes interference. We are currently working on an Android library to use ultrasound as a second signal way for our *opptain* application and other Android-based projects.

Messages in *Opportunistic Networks* have an expiry date called *Time-To-Live* and when a message is expired, it will be dropped from the buffer. However, messages in the buffer, no matter if expired or not, do not disturb any functionality of the network as long as the buffer is not full. These messages, of course, should not be forwarded to relay nodes anymore since that would change the originally planned order of the message queue and routing protocols would not be followed. These messages could stay in the buffer, however, for a direct delivery to the destination. If the intended destination of a message is interested in the message and there is a direct contact between these nodes, the message can still be delivered from the buffer. We have already conducted some preliminary tests that show that we can increase the delivery ratio in the network by about 2% when keeping expired messages in the buffer and, at the earliest, delete them when the buffer is full, even before applying the used drop policy. For future work, it is desirable to test this principle with multiple routing protocols, drop policies and simulation scenarios to see if the delivery ratio can be increased with this method. Additionally, it has to be discussed in what way the *Time-To-Live* is used for a message. If an application sets a *Time-To-Live* because of an application-related expiry of the message, keeping it in the buffer might be useless. However, if the *Time-To-Live* is network-based, i.e., set by the network to reduce transmission overhead in the network, keeping the message in the buffer does no harm and applications can profit from our approach.

We conducted multiple experiments with our *opptain* application including a field experiment with a fairly attractive node density. While we provide the already mentioned messaging and filesharing apps, they have not been tested in these experiments. In the course of this dissertation and on top of the publications, we also provide a gaming app framework for round-based games. To further test the *opptain* network, it would be beneficial to find a group of people in a dense environment willing to test these user applications. These tests could not just be conducted by using the app with the purpose of evaluation but simply by using it in an everyday fashion. This might be possible by handing out the *opptain* app and the user apps to (grassroots) initiatives and local communities like *Freifunk* that care about free Internet or net neutrality and might be interested in participating in a trial of applying *Opportunistic Networks* in a real-world setting.

With the creation of our *opptain* framework we laid the foundation for smartphone-based *Opportunistic Networks*. The distribution and acceptance of our framework in a community is the next goal that we want to accomplish. Long-term, we hope that *Opportunistic Networks* and their benefits will become more well-known and used across the world.

Glossary

ansWEARs *ansWEARs* is a for easy communication between smartphone and smartwatch that also offers the exchange of various information that the smartwatch is collecting about the network and its physical surroundings. 13, 49, 71

API Application Programming Interface. An *Application Programming Interface* is a specification of how software components have to interact. 7, 21, 23, 50, 69, 70, 73

Bluetooth *Bluetooth* is a wireless technology standard for short range communication. On early Android versions, *Bluetooth* devices have to be paired which requires user interaction. 1, 18–20, 32, 33, 44, 50, 51, 58, 71, 74, 77

Bluetooth Low Energy *Bluetooth Low Energy* is a wireless technology standard for short range communication that reduces the power consumption of classic *Bluetooth*. 19, 71, *see* Bluetooth

communication island A *communication island* is an isolated subgraph in a network whose topology is otherwise highly dynamic. 18, 41, 43–45, 74

Delay Tolerant Network A *Delay Tolerant Network* is a challenged network which can experience frequent and prolonged partitioning and in which nodes may not have an end-to-end path. This can happen due to node mobility, changes in signal strength or because of a periodic or predictable behavior [34, 41]. 32, 81, 83

delivery ratio The *delivery ratio* is one of the three important metrics used in *Opportunistic Networks* to make a statement about the quality. This metric is defined as the ratio between the number of received messages and the number of generated messages. The

aim of all components of *Opportunistic Networks* is to maintain a high *delivery ratio* in the network. 11, 30, 33, 39–41, 54, 55, 57–59, 61–65

drop policy A *drop policy* defines which message in the buffer has to be dropped if the buffer is full and a new message is added. Because these *drop policies* define a order of messages, the buffer is considered a queue which is why drop policies are also called *queueing policies* [33]. We will use the term *drop policy* for this thesis explicitly for dropping and use the term *queueing policy* as a generic term for a policy that orders a forward or drop queue. Examples for *drop policies* are *First-In-First-Out (FIFO)* which uses the minimum arrival time of a message as the drop criteria, *Evict-Shortest-Lifetime-First (SHLI)* which uses the maximum *Time-To-Live* as the drop criteria or *Evict-Most-Forwarded-First (MOFO)* which uses the maximum *replication count* as the main decision criteria and the *hop count* as a tie breaker [33]. 8, 14, 17, 24, 26–28, 55–59, 61–64, 72, 73, 78, 79

end-to-end delay The *end-to-end delay* is one of the three important metrics used in *Opportunistic Networks* to make a statement about the quality. This metric is defined as the duration between the creation of a message and its arrival at its destination, averaged over all messages that reach their destination. The aim of all components of *Opportunistic Networks* is to maintain a low *delay* in the network for the user applications that prefer a fast delivery of their messages. We will mostly use the term *delay* as short form. 11, 30, 33, 39–41, 54, 55, 59, 61–63, 65

Epidemic Epidemic is a simple flooding-based routing protocol for intermittently connected networks. Vahdat and Becker present this protocol in which nodes replicate a message to all connected nodes which previously did not have a copy of the message [48]. 14, 31, 32, 40, 41, 53–59, 61–66, 80, 81

Evict-Most-Forwarded-First (MOFO) *Evict Most Forwarded First (MOFO)* [33] is a *drop policy* which uses the maximum *replication count* as the main decision criteria and the *hop count* as a tie breaker. 56, 59, 78, *see* drop policy

Evict-Shortest-Lifetime-First (SHLI) *Evict-Shortest-Lifetime-First (SHLI)* [33] is a *drop policy* which uses the maximum *Time-To-Live* as drop criteria. 78, *see* drop policy

First-In-First-Out (FIFO) *First-In-First-Out (FIFO)*, also *first come, first served* [33], is a policy which uses the minimum arrival time of a message as the queueing criteria. This

policy can be used as a *drop policy* and for ordering the queue of a *forward strategy*. 14, 55, 56, 59, 61, 62, 72, 73, 78, *see* drop policy & forward strategy

forward strategy After two connected nodes exchanged meta information, they have to decide which messages to forward to the connected node. A *forward strategy* defines if a message should be forwarded to the currently connected node or not [33]. Furthermore, *forward strategies* may also define the order in which the messages should be forwarded. The order is important as unexpected interruptions may occur and a node cannot forward all messages it would like to [33]. *GRTR*, for example, is a *forward strategy* that forwards a message only if the delivery predictability for the message's destination is higher at the connected node [33]. 4, 8, 14, 17, 54–59, 61–63, 72, 79, 82

GPRS General Packet Radio Service. The *General Packet Radio Service* is a mobile data standard for cellular communication. 1, 83

GPS Global Positioning System. The *Global Positioning System* is navigation satellite system which offers location and time data to a *GPS* receiver. 1, 26, 27, 72

hop count The *hop count* of a message is the number of times the message was forwarded on the way from source to destination. The *hop count* is both a router metric that is used for decisions and a performance metric that is used for evaluation. 14, 15, 40, 53, 54, 57–59, 61, 63–65, 72, 73, 78

Hotspotting On Android smartphones, users can open a *Wi-Fi* access point (see *Infrastructure Mode Wi-Fi*) to share the Internet connection with other devices; these access points are called tethering hotspots. Since ad hoc *Wi-Fi* (see *IBSS*) is not available on Android smartphones, we use the following mechanism, called *hotspotting*, to connect smartphones with each other automatically and without user interaction. The device scans for tethering hotspots of other network participants in the surroundings and, after a successful search, connects to one. If no hotspot is nearby, the device itself becomes a hotspot and lets other devices connect to it. 13, 15, 20, 22, 43, 45, 46, 50, 69, 71, *see* IBSS

IBSS An *IBSS (Independent Basic Service Set)* or ad hoc network is an IEEE 802.11 network without infrastructure which consists of two or more nodes that communicate with each other directly. Because this *IBSS* mode is not specified in detail, these ad hoc networks

of different manufacturers might not be able to communicate with each other. 19, 73, 79, 80, *see* Infrastructure Mode Wi-Fi

Infrastructure Mode Wi-Fi With *Infrastructure Mode Wi-Fi* which is defined by IEEE 802.11, devices connect wirelessly to an access point. The other mode, the ad hoc (*IBSS*) mode, can be used to connect devices directly to each other. 7, 12, 18, 19, 47, 79, *see* IBSS

Intent Intents are Android messages used for intra- or inter-application communication. 36, 37

Inter-Contact Time The *Inter-Contact Time* is the time between node meetings and has a high impact on the delay in the network. 33, 55, 58, 59

IPND Internet Protocol Neighbor Discovery. Nodes that use the *Internet Protocol Neighbor Discovery* send and listen for announcements that can be used to learn of the existence of other network participants [49]. 33

LTE Long Term Evolution. *Long Term Evolution* is a high-speed mobile data standard for cellular communication. 1, 83

LTE Direct *LTE Direct* is a device-to-device technology which enables mobile devices to discover each other and interact. 18, 19, *see* LTE

Markov Chain model A Markov Chain model is a "stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event" [50]. Groenevelt et al. [35] modeled *Epidemic* and two-hop routing using Markov chains, which we use for our system models. 58, 62

MOST-RPT *MOST-RPT (Most Of Storage and Transmission – with Replication Probability Threshold)* is a replication scheme proposed in our paper *Dynamic Replication Control Strategy for Opportunistic Networks* [11]. 14, 55–59, 61, 62, 64, 65, 72, 73

NFC Near Field Communication. *Near Field Communication* is a short-range wireless communication technology. 1, 18, 19

NTP Network Time Protocol. The *Network Time Protocol* is a protocol for clock synchronization in data networks. 9, 25–27, 69

Opportunistic Network An *Opportunistic Network* is a subclass of a *Delay Tolerant Network* with intermittent node connections in which messages are routed from source to destination via *Store-Carry-Forward* because there is no end-to-end path between source and destination [1, 51]. 2–15, 17–26, 28, 29, 31, 32, 35, 36, 39–41, 43–46, 53, 54, 56–59, 61–65, 67–71, 73–75, 77, 78, 81–83

opptain *opptain* is the Android-based *Opportunistic Network* application developed in this dissertation. 8–15, 18–25, 27, 29–32, 35–37, 39–41, 43, 45–51, 56, 68–75

ORBOPH *Optimal Replication Based on Optimal Path Hops (ORBOPH)* is a replication scheme proposed in our paper *Optimal Replication Based on Optimal Path Hops for Opportunistic Networks* [13]. 14, 15, 55, 56, 64–66, 73

Ordinary Differential Equation An *Ordinary Differential Equation (ODE)* is an equation that contains a function of exactly one independent variable and its derivatives. Zhang et al. [36] derived a framework based on *Ordinary Differential Equations* to analyze the routing protocol *Epidemic*. 58, 62

overhead The *overhead* is one of the three important metrics used in *Opportunistic Networks* to make a statement about the quality. This metric is defined as the average number of copies of a message that has been delivered to its destination. The aim of all components of *Opportunistic Networks* is to maintain a low *overhead* in the network because a high *overhead* induces a higher consumption of storage space and bandwidth which can negatively affect the delivery of other messages. 11, 30, 39–41, 54, 55, 57–59, 61–65

PRoPHET The *Probabilistic Routing Protocol using History of Encounters and Transitivity*, short *PRoPHET*, is a *routing protocol* which maintains message delivery probabilities. Nodes forward the message only if the connected node has a higher delivery predictability [52]. An attempt to increase security in *PRoPHET* can be found in [53]. 40, 41, 54, 56, 58, *see* routing protocol

QR code Quick Response Code. A *Quick Response Code* is a two-dimentional type of barcode. 69

queueing policy A *queueing policy* defines in which order messages are forwarded. 14, 17, 55, 59, 61, 62, 64, 65, 78

replication count The *replication count* of a message at one specific node is the number of times the message was replicated at this node. 14, 15, 54, 57–59, 61, 63–65, 72, 73, 78

rooted By *rooting* an Android smartphone, the user obtains root access to the device. *Rooting* disables Android's security architecture which is why Android devices usually come *unrooted*. 7, 18–20, 36, 45

routing protocol In general, a *routing protocol* specifies which path a node uses to transport a message through the network. In *Opportunistic Networks*, in which there is no end-to-end path between the source and destination of messages, the routing protocol specifies which information is saved at each node and in the message header, which information is shared with connected nodes and which *forward strategy* is (or can be) used. 4, 8, 14, 15, 17, 20, 22, 30, 53, 54, 56, 59, 62, 63, 72, 81, 82, *see* forward strategy

RPRS *Replication Probability-based Routing Scheme (RPRS)* is a replication scheme proposed in our paper of the same name [12]. 14, 55, 56, 61–65, 73

Spray & Wait *Spray & Wait* is a *routing protocol* consisting the spray phase and the wait phase. The source node has a fixed number of copies of the message and is allowed to *spray* these copies to distinct nodes. These nodes, however, are only allowed to *wait* until they connect to the destination of the message to forward the copy. In a second version, called *Binary Spray & Wait*, the nodes in the *spray* phase are allowed to forward half of their copies to other nodes; only if a node has one copy left, it enters the *wait* phase. 54, 55, 61, 63, *see* routing protocol

SSID Service Set Identifier. The *Service Set Identifier* is an up to 32 character long name of an 802.11 *Wi-Fi* access point which is broadcast to help clients find the associated network. 47

Store-Carry-Forward *Store-Carry-Forward* is an extension of the *Store and Forward* technique. With *Store and Forward*, messages are stored at a router or node and kept there until they are forwarded to the destination or another router on the path to the destination. In *Opportunistic Networks*, nodes are mobile devices that move around and carry the

message to the next hop, thus the technique is called *Store-Carry-Forward*. 2, 4, 9, 10, 32, 53, 57, 58, 62, 81

the ONE simulator The *ONE* simulator, short for *Opportunistic Networking Environment*, is a Java-based simulation framework for *Delay Tolerant Networks* and *Opportunistic Networks*. It offers a wide range of protocols, mobility models, and event generators, and allows developers to extend all components to their wishes [54]. 33, 59, 65, 71, 73

Time-To-Live The *Time-To-Live* is a value to limit the existence of messages in the network; if the *Time-To-Live* of a message is expired, the message is dropped from the node's buffer. In traditional networks, the *Time-To-Live* is the number of hops the message can travel trough the network before it is dropped. Because in *Opportunistic Networks* it is desired that a large number of nodes carry the message trough the network, we use real time to define our *Time-To-Live* and let messages expire after a span of time. 5, 9, 24–26, 30, 40, 41, 55, 58, 59, 63, 65, 69, 74, 78

UMTS Universal Mobile Telecommunications System. The *Universal Mobile Telecommunications System* is a mobile data standard for cellular communication with a transfer rate higher than *GPRS* but lower than *LTE*. 1

Wi-Fi *Wi-Fi* is a wireless technology for local area networking based on the IEEE 802.11 standards. 1, 13, 18–20, 22, 32, 33, 50, 58, 71, 72, 74, 79, 82, 83

Wi-Fi Aware *Wi-Fi Aware* is a *Wi-Fi* standard that enables devices to discover and connect to each other directly, and to exchange data without creating a network connection [55]. 73, 74, *see* IBSS

Bibliography

[1] Luciana Pelusi, Andrea Passarella, and Marco Conti. "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks". In: *IEEE Communications Magazine* 44.11 (2006), pp. 134–141. DOI: 10.1109/MCOM.2006.248176. URL: https://doi.org/10.1109/MCOM.2006.248176 (Pages: 2, 81).

[2] Inc. Gartner. *Gartner Says Worldwide Sales of Smartphones Returned to Growth in First Quarter of 2018*. https://www.gartner.com/newsroom/id/3876865. [Online]. Last access on: 2018-11-22. 2018 (Pages: 4, 7, 18).

[3] Salem Sati. "Efficient Connection Establishment, Message Handling and Content Delivery in Opportunistic Networks". PhD thesis. University of Düsseldorf, Germany, 2017. URL: https://docserv.uni-duesseldorf.de/servlets/DocumentServlet?id=42917 (Pages: 8, 11, 17, 28).

[4] Andre Ippisch and Kalman Graffi. "Infrastructure Mode Based Opportunistic Networks on Android Devices". In: *Proceedings of the 31st IEEE International Conference on Advanced Information Networking and Applications, AINA 2017, Taipei, Taiwan, March 27-29, 2017*. 2017, pp. 454–461. DOI: 10.1109/AINA.2017.32. URL: https://doi.org/10.1109/AINA.2017.32 (Pages: 15, 17, 21).

[5] Andre Ippisch, Tobias Kuper, and Kalman Graffi. "Time and Space in Android-Based Opportunistic Networks". In: *Proceedings of the 32nd IEEE International Conference on Advanced Information Networking and Applications, AINA 2018, Krakow, Poland, May 16-18, 2018*. 2018, pp. 244–250. DOI: 10.1109/AINA.2018.00046. URL: https://doi.org/10.1109/AINA.2018.00046 (Pages: 15, 24).

[6] Andre Ippisch, Jannik Leßenich, and Kalman Graffi. "Contact Matching and Connection Scheduling in Android-based Opportunistic Networks". In: *Proceedings of the 15th Annual Conference on Wireless On-demand Network Systems and Services, WONS 2019, Wengen, Switzerland, January 22-24, 2019*. 2019 (Pages: 15, 45).

[7] Bashkim Berzati, Andre Ippisch, and Kalman Graffi. "An Android Wear OS Framework for Sensor Data and Network Interfaces". In: *Proceedings of the 43rd IEEE Conference on Local Computer Networks Workshops, LCN 2018, Chicago, USA, October 1-4, 2018*. 2018 (Pages: 15, 43, 49).

[8] Andre Ippisch, Salem Sati, and Kalman Graffi. "Device to device communication in mobile Delay Tolerant networks". In: *Proceedings of the 21st IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications, DS-RT 2017, Rome, Italy, October 18-20, 2017*. 2017, pp. 91–98. DOI: 10.1109/DISTRA.2017.8167671. URL: https://doi.org/10.1109/DISTRA.2017.8167671 (Pages: 15, 29, 31).

[9] Andre Ippisch, Martin Nowak, and Kalman Graffi. "Mitigation Techniques for Software- and Network-based Threats in Android-based Opportunistic Networks". In: *Proceedings of the 15th Annual Conference on Wireless On-demand Network Systems and Services, WONS 2019, Wengen, Switzerland, January 22-24, 2019*. 2019 (Pages: 15, 29, 35).

[10] Andre Ippisch, Philipp Brühn, and Kalman Graffi. "Field Experiment on the Performance of an Android-based Opportunistic Network". In: *Proceedings of the International Euro-Par 2018 Parallel Processing Workshops, Turin, Italy, August 27-31, 2018*. 2018 (Pages: 15, 30, 39).

[11] Salem Sati, Andre Ippisch, and Kalman Graffi. "Dynamic replication control strategy for Opportunistic Networks". In: *Proceedings of the 2017 International Conference on Computing, Networking and Communications, ICNC 2017, Silicon Valley, CA, USA, January 26-29, 2017*. 2017, pp. 1017–1023. DOI: 10.1109/ICCNC.2017.7876274. URL: https://doi.org/10.1109/ICCNC.2017.7876274 (Pages: 16, 54, 57, 61, 64, 80).

[12] Salem Sati, Andre Ippisch, and Kalman Graffi. "Replication probability-based routing scheme for opportunistic networks". In: *Proceedings of the 2017 International Conference on Networked Systems, NetSys 2017, Göttingen, Germany, March 13-16, 2017*. 2017, pp. 1–8. DOI: 10.1109/NetSys.2017.7903953. URL: https://doi.org/10.1109/NetSys.2017.7903953 (Pages: 16, 54, 55, 61, 64, 82).

[13] Andre Ippisch, Salem Sati, and Kalman Graffi. "Optimal Replication Based on Optimal Path Hops for Opportunistic Networks". In: *Proceedings of the 32nd IEEE International Conference on Advanced Information Networking and Applications, AINA 2018, Krakow, Poland, May 16-18, 2018*. 2018, pp. 251–258. DOI: 10.1109/AINA.2018.00047. URL: https://doi.org/10.1109/AINA.2018.00047 (Pages: 16, 54, 55, 64, 81).

[14] Thinktube Inc. *Thinktube - WiFi IBSS*. http://www.thinktube.com/index.php/tech-en/android/wifi-ibss. [Online]. Last access on: 2018-11-22. 2018 (Page: 19).

[15] Google. *Android Open Source Project*. https://source.android.com/. [Online]. Last access on: 2018-11-22. 2018 (Page: 19).

[16] Sacha Trifunovic, Maciej Kurant, Karin Anna Hummel, and Franck Legendre. "WLAN-Opp: Ad-hoc-less opportunistic networking on smartphones". In: *Ad Hoc Networks* 25 (2015), pp. 346–358. DOI: 10.1016/j.adhoc.2014.07.011. URL: https://doi.org/10.1016/j.adhoc.2014.07.011 (Page: 19).

[17] The Serval Project. *The Serval Project*. http://www.servalproject.org/. [Online]. Last access on: 2018-11-22. 2018 (Page: 20).

[18] Briar. *Briar*. https://briarproject.org/. [Online]. Last access on: 2018-11-22. 2018 (Page: 20).

[19] Open Garden. *Mesh networking made easy - Open Garden*. https://www.opengarden.com. [Online]. Last access on: 2018-11-22. 2018 (Page: 20).

[20] Bridgefy. *Bridgefy: The SDK that makes apps work offline*. https://www.bridgefy.me/. [Online]. Last access on: 2018-11-22. 2018 (Page: 20).

[21] Johannes Morgenroth, Sebastian Schildt, and Lars C. Wolf. "A bundle protocol implementation for android devices". In: *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking, Mobicom'12, Istanbul, Turkey, August 22-26, 2012*. 2012, pp. 443–446. DOI: 10.1145/2348543.2348606. URL: http://doi.acm.org/10.1145/2348543.2348606 (Page: 20).

[22] Keith Scott and Scott Burleigh. *Bundle Protocol Specification*. RFC 5050. RFC Editor, 2007. URL: http://www.rfc-editor.org/rfc/rfc5050.txt (Page: 20).

[23] Andre Ippisch and Kalman Graffi. "An Android Framework for Opportunistic Wireless Mesh Networking". In: *Proceedings of the 2015 International Conference on Networked Systems, NetSys 2015, Cottbus, Germany, March 9-13, 2015*. 2015 (Page: 21).

[24] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li-Shiuan Peh, and Daniel Rubenstein. "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet". In: *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X), San Jose, California, USA, October 5-9, 2002*. 2002, pp. 96–107. DOI: 10.1145/605397.605408. URL: http://doi.acm.org/10.1145/605397.605408 (Page: 24).

[25] Anders Lindgren, Avri Doria, Jan Lindblom, and Mattias Ek. "Networking in the land of northern lights: two years of experiences from DTN system deployments". In: *Proceedings of the 2008 ACM Workshop on Wireless Networks and Systems for Developing Regions, San Francisco, California, USA, September 19, 2007*. 2008, pp. 1–8. DOI: 10.1145/1410064.1410066. URL: http://doi.acm.org/10.1145/1410064.1410066 (Page: 24).

[26] David L. Mills, Jim Martin, Jack Burbank, and William Kasch. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. RFC 5905. RFC Editor, 2010. URL: https://rfc-editor.org/rfc/rfc5905.txt (Page: 27).

[27] Ken Behrendt and Ken Fodero. "The Perfect Time: An Examination of Time-Synchronization Techniques". In: *Schweitzer Engineering Laboratories* (2006) (Page: 27).

[28] Bong Jun Choi and Xuemin Shen. "Distributed Clock Synchronization in Delay Tolerant Networks". In: *Proceedings of the IEEE International Conference on Communications, ICC 2010, Cape Town, South Africa, 23-27 May 2010*. 2010, pp. 1–6. DOI: 10.1109/ICC.2010.5502781. URL: https://doi.org/10.1109/ICC.2010.5502781 (Page: 27).

[29] Dong Zhou and Ten-Hwang Lai. "An Accurate and Scalable Clock Synchronization Protocol for IEEE 802.11-Based Multihop Ad Hoc Networks". In: *IEEE Transactions on Parallel and Distributed Systems* 18.12 (2007), pp. 1797–1808. DOI: 10.1109/TPDS.2007.1116. URL: https://doi.org/10.1109/TPDS.2007.1116 (Page: 27).

[30] Qun Li and Daniela Rus. "Global Clock Synchronization in Sensor Networks". In: *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2004, Hong Kong, China, March 7-11, 2004*. 2004. DOI: 10.1109/INFCOM.2004.1354528. URL: https://doi.org/10.1109/INFCOM.2004.1354528 (Page: 27).

[31] Philipp Sommer and Roger Wattenhofer. "Gradient clock synchronization in wireless sensor networks". In: *Proceedings of the 8th International Conference on Information Processing in Sensor Networks, IPSN 2009, April 13-16, 2009, San Francisco, California, USA*. 2009, pp. 37–48. DOI: 10.1145/1602165.1602171. URL: http://doi.acm.org/10.1145/1602165.1602171 (Page: 27).

[32] Salem Sati, Christopher Probst, and Kalman Graffi. "Analysis of Buffer Management Policies for Opportunistic Networks". In: *Proceedings of the 25th International Conference on Computer Communication and Networks, ICCCN 2016, Waikoloa, HI, USA, August 1-4, 2016*. 2016, pp. 1–8. DOI: 10.1109/ICCCN.2016.7568494. URL: https://doi.org/10.1109/ICCCN.2016.7568494 (Page: 28).

[33] Anders Lindgren and Kaustubh S. Phanse. "Evaluation of queueing policies and forwarding strategies for routing in intermittently connected networks". In: *Proceedings of the First International Conference on COMmunication System softWAre and MiddlewaRE, COMSWARE 2006, January 8-12, 2006, New Delhi, India*. 2006, pp. 1–10. DOI: 10.1109/COMSWA.2006.1665196. URL: https://doi.org/10.1109/COMSWA.2006.1665196 (Pages: 28, 78, 79).

[34] Kevin R. Fall. "A delay-tolerant network architecture for challenged internets". In: *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM 2003, August 25-29, 2003, Karlsruhe, Germany*. 2003, pp. 27–34. DOI: 10.1145/863955.863960. URL: http://doi.acm.org/10.1145/863955.863960 (Pages: 33, 77).

[35] Robin Groenevelt, Philippe Nain, and Ger Koole. "The message delay in mobile ad hoc networks". In: *Performance Evaluation* 62.1-4 (2005), pp. 210–228. DOI: 10.1016/j.peva.2005.07.018. URL: https://doi.org/10.1016/j.peva.2005.07.018 (Pages: 33, 80).

[36] Xiaolan Zhang, Giovanni Neglia, James F. Kurose, and Donald F. Towsley. "Performance modeling of epidemic routing". In: *Computer Networks* 51.10 (2007), pp. 2867–2891. DOI: 10.1016/j.comnet.2006.11.028. URL: https://doi.org/10.1016/j.comnet.2006.11.028 (Pages: 33, 81).

[37] Suzan Bayhan, Esa Hyytiä, Jussi Kangasharju, and Jörg Ott. "Analysis of hop limit in opportunistic networks by static and time-aggregated graphs". In: *Proceedings of the 2015 IEEE International Conference on Communications, ICC 2015, London, United Kingdom, June 8-12, 2015*. 2015, pp. 3287–3292. DOI: 10.1109/ICC.2015.7248831. URL: https://doi.org/10.1109/ICC.2015.7248831 (Page: 33).

[38] Muhammad Abdulla and Robert Simon. "The impact of intercontact time within opportunistic networks: protocol implications and mobility models". In: *TechRepublic White Paper* (2009) (Page: 33).

[39] Ahmad Cheraghi, Tobias Amft, Salem Sati, Philipp Hagemeister, and Kalman Graffi. "The State of Simulation Tools for P2P Networks on Mobile Ad-Hoc and Opportunistic Networks". In: *Proceedings of the 25th International Conference on Computer Communication and Networks, ICCCN 2016, Waikoloa, HI, USA, August 1-4, 2016*. 2016, pp. 1–7. DOI: 10.1109/ICCCN.2016.7568584. URL: https://doi.org/10.1109/ICCCN.2016.7568584 (Pages: 40, 74).

[40] Jannik Leßenich. "Topology Control in Android-based Opportunistic Networks". Master's Thesis. Department of Computer Science, Heinrich Heine University Düsseldorf, Sept. 2018 (Page: 44).

[41] Sushant Jain, Kevin R. Fall, and Rabin K. Patra. "Routing in a delay tolerant network". In: *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM 2004, August 30 - September 3, 2004, Portland, Oregon, USA*. 2004, pp. 145–158. DOI: 10.1145/1015467.1015484. URL: http://doi.acm.org/10.1145/1015467.1015484 (Pages: 53, 77).

[42] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta G. Jetcheva. "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols". In: *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking, MOBICOM '98, Dallas, Texas, USA, October 25-30, 1998*. 1998, pp. 85–97. DOI: 10.1145/288235.288256. URL: http://doi.acm.org/10.1145/288235.288256 (Page: 53).

[43] Thrasyvoulos Spyropoulos, Rao Naveed Bin Rais, Thierry Turletti, Katia Obraczka, and Athanasios V. Vasilakos. "Routing for disruption tolerant networks: taxonomy and design". In: *Wireless Networks* 16.8 (2010), pp. 2349–2370. DOI: 10.1007/s11276-010-0276-9. URL: https://doi.org/10.1007/s11276-010-0276-9 (Page: 53).

[44] Tobias Amft, Barbara Guidi, Kalman Graffi, and Laura Ricci. "FRoDO: Friendly routing over dunbar-based overlays". In: *Proceedings of the 40th IEEE Conference on Local Computer Networks, LCN 2015, Clearwater Beach, FL, USA, October 26-29, 2015*. 2015, pp. 356–364. DOI: 10.1109/LCN.2015.7366330. URL: https://doi.org/10.1109/LCN.2015.7366330 (Page: 74).

[45] Tobias Amft and Kalman Graffi. "Moving peers in distributed, location-based peer-to-peer overlays". In: *Proceedings of the 2017 International Conference on Computing, Networking and Communications, ICNC 2017, Silicon Valley, CA, USA, January 26-29, 2017*. 2017, pp. 906–911. DOI: 10.1109/ICCNC.2017.7876253. URL: https://doi.org/10.1109/ICCNC.2017.7876253 (Page: 74).

[46] Kalman Graffi. "Monitoring and Management of Peer-to-Peer Systems". PhD thesis. Technische Universität Darmstadt, 2010. ISBN: 978-3-86853-658-4. URL: tuprints.ulb.tu-darmstadt.de/2248/ (Page: 74).

[47] Matthias Feldotto and Kalman Graffi. "Comparative evaluation of peer-to-peer systems using PeerfactSim.KOM". In: *Proceedings of the International Conference on High Performance Computing & Simulation, HPCS 2013, Helsinki, Finland, July 1-5, 2013*. 2013, pp. 99–106. DOI: 10.1109/HPCSim.2013.6641399. URL: https://doi.org/10.1109/HPCSim.2013.6641399 (Page: 74).

[48] Amin Vahdat and David Becker. *Epidemic Routing for Partially-Connected Ad Hoc Networks*. Technical Report. 2000. URL: http://www.cs.duke.edu/techreports/2000/2000-06.ps (Page: 78).

[49] Daniel Ellard and Richard Altmann. *DTN IP Neighbor Discovery (IPND)*. Internet-Draft. 2010. URL: https://tools.ietf.org/html/draft-irtf-dtnrg-ipnd-01 (Page: 80).

[50] Oxford Dictionaries. *Markov chain | Definition*. https://en.oxforddictionaries.com/definition/us/markov_chain. [Online]. Last access on: 2018-11-22. 2018 (Page: 80).

[51] Chung-Ming Huang, Kun-Chan Lan, and Chang-Zhou Tsai. "A Survey of Opportunistic Networks". In: *Proceedings of the 22nd International Conference on Advanced Information Networking and Applications, AINA 2008, Workshops Proceedings, GinoWan, Okinawa, Japan, March 25-28, 2008*. 2008, pp. 1672–1677. DOI: 10.1109/WAINA.2008.292. URL: https://doi.org/10.1109/WAINA.2008.292 (Page: 81).

[52] Anders Lindgren, Avri Doria, and Olov Schelén. "Probabilistic Routing in Intermittently Connected Networks". In: *Proceedings of the First International Workshop on Service Assurance with Partial and Intermittent Resources, SAPIR 2004, Fortaleza, Brazil, August 1-6, 2004*. 2004, pp. 239–254. DOI: 10.1007/978-3-540-27767-5_24. URL: https://doi.org/10.1007/978-3-540-27767-5_24 (Page: 81).

[53] Raphael Bialon and Kalman Graffi. "Misrouted Prophecy - On the Impact of Security Attacks on PRoPHET". In: *Proceedings of the International Euro-Par 2016 Parallel Processing Workshops, Grenoble, France, August 24-26, 2016, Revised Selected Papers*. 2016, pp. 296–308. DOI: 10.1007/978-3-319-58943-5_24. URL: https://doi.org/10.1007/978-3-319-58943-5_24 (Page: 81).

[54] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. "The ONE simulator for DTN protocol evaluation". In: *Proceedings of the 2nd International Conference on Simulation Tools and Techniques for Communications, Networks and Systems, SimuTools 2009, Rome, Italy, March 2-6, 2009*. 2009, p. 55. DOI: 10.4108/ICST.SIMUTOOLS2009.5674. URL: https://doi.org/10.4108/ICST.SIMUTOOLS2009.5674 (Page: 83).

[55] Wi-Fi Alliance. *Wi-Fi Aware | Wi-Fi Alliance*. https://www.wi-fi.org/discover-wi-fi/wi-fi-aware. [Online]. Last access on: 2018-11-22. 2015 (Page: 83).

Personal Publications

[P1] Andre Ippisch and Kalman Graffi. “An Android Framework for Opportunistic Wireless Mesh Networking”. In: *Proceedings of the GI/ITG International Conference on Networked Systems (NetSys)*. 2015.

[P2] Kalman Graffi and Andre Ippisch. “Accelerating Data Synchronization between Smartphones and Tablets using PowerFolder in IEEE 802.11 Infrastructure-based Mesh Networks”. In: *Proceedings of the Qatar Foundation Annual Research Conference (ARC)*. 2016.

[P3] Salem Sati, Andre Ippisch and Kalman Graffi. “Dynamic Replication Control Strategy for Opportunistic Networks”. In: *Proceedings of the IEEE International Conference on Computing, Networking and Communications (ICNC)*. 2017.

[P4] Salem Sati, Andre Ippisch and Kalman Graffi. “Replication Probability-based Routing Scheme for Opportunistic Networks”. In: *Proceedings of the GI/ITG International Conference on Networked Systems (NetSys)*. 2017.

[P5] Andre Ippisch and Kalman Graffi. “Infrastructure Mode Based Opportunistic Networks on Android Devices”. In: *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA)*. 2017.

[P6] Andre Ippisch, Salem Sati and Kalman Graffi. “Device to Device Communication in Mobile Delay Tolerant Networks”. In: *Proceedings of the IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications (DS-RT)*. 2017.

[P7] Andreas Disterhöft, Phillip Sandkühler, Andre Ippisch and Kalman Graffi. “Mr.Tree: Multiple Realities in Tree-based Monitoring Overlays for Peer-to-Peer Networks”. In: *Proceedings of the IEEE International Conference on Computing, Networking and Communications (ICNC)*. 2018.

[P8] Andre Ippisch, Tobias Küper and Kalman Graffi. "Time and Space in Android-based Opportunistic Networks". In: *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA)*. 2018.

[P9] Andre Ippisch, Salem Sati and Kalman Graffi. "Optimal Replication Based on Optimal Path Hops for Opportunistic Networks". In: *Proceedings of the IEEE International Conference on Advanced Information Networking and Applications (AINA)*. 2018.

[P10] Andre Ippisch, Philipp Brühn and Kalman Graffi. "Field Experiment on the Performance of an Android-based Opportunistic Network". In: *Proceedings of the International Conference on Parallel and Distributed Computing (Euro-Par)*. 2018.

[P11] Bashkim Berzati, Andre Ippisch and Kalman Graffi. "An Android Wear OS Framework for Sensor Data and Network Interfaces". In: *Proceedings of the IEEE International Conference on Local Computer Networks (LCN)*. 2018.

[P12] Andre Ippisch, Jannik Leßenich and Kalman Graffi. "Contact Matching and Connection Scheduling in Android-based Opportunistic Networks". *Technical Report: TR-2018-001*. Technology of Social Networks Group, Heinrich Heine University, Düsseldorf, Germany. 2018.

[P13] Andre Ippisch, Martin Nowak and Kalman Graffi. "Mitigation Techniques for Software- and Network-based Threats in Android-based Opportunistic Networks". *Technical Report: TR-2018-002*. Technology of Social Networks Group, Heinrich Heine University, Düsseldorf, Germany. 2018.