Opportunistic Content Distribution

ÓLAFUR RAGNAR HELGASON

Doktorsavhandling
Stockholm, Sverige 2011
Opportunistic Content Distribution

ÓLAFUR RAGNAR HELGASON

Doctoral Thesis
Stockholm, Sweden 2011
Abstract

In recent decades, communication networks have had a profound effect on society. Wireless communication has affected our lifestyle and altered how humans communicate and the Internet has revolutionized how we access, publish and disseminate information. In recent years we have also witnessed a radical change in how information is generated on the Internet. Today, information is no longer only generated by a small group of professionals but it is created by the users themselves and shared with a broad community with matching interests. This is evident with "Web 2.0" applications such as blogs, podcasts, YouTube and social platforms like Facebook and Flickr. As a result of these trends, the Internet is today mainly used to provide users with access to contents. With recent advances in mobile platforms, information generation and consumption has spread from personal computers and Internet into people's palms. This calls for efficient dissemination of information to and from mobile devices.

This thesis considers content-centric networking in the context of mobile wireless networks. The main focus is on opportunistic distribution of content where mobile nodes directly exchange content items when they are within communication range. This communication mode enables dissemination of content between mobile nodes without relying on infrastructure, which can be beneficial for several reasons: infrastructure may be absent, overloaded, unreliable, expensive to use, censored or limited to certain users or contents. Opportunistic networking also has different properties than infrastructure based wireless networking, particularly in terms of scalability, locality and dissemination delay.

The contributions of this thesis lie in two areas. Firstly we study the feasibility and performance of opportunistic networking among mobile nodes in urban areas using both analytic models and simulations. In particular we study the effect of two enablers of opportunistic networking: cooperation and mobility. By applying models from epidemic modeling, we show that if nodes cooperate by sharing, even in a limited manner, content can spread efficiently in a number of common case scenarios. We also study in detail which aspects of human mobility affect wireless communication and conclude that performance is not very sensitive to accurate estimation of the probability distributions of mobility parameters such as speed and arrival process. Our results however suggest that it is important to capture the scenario and space in which mobility occurs since this may affect performance significantly. Secondly, we present our design and implementation of a middleware architecture for a mobile peer-to-peer content distribution. Our system uses a decentralized content solicitation scheme that allows the distribution of content between mobile devices without requiring Internet connectivity and infrastructure support. Our system is based on the publish/subscribe paradigm and we describe the design and implementation of key components. We evaluate the performance and correctness of the system using both large-scale simulations and small-scale experimentation with our implementation. Finally we present the design and evaluation of an energy-efficient radio subsystem for opportunistic networking.
Acknowledgments

First of all I would like to thank my advisor, Prof. Gunnar Karlsson, for giving me the opportunity to join his lab and for his support, encouragement and guidance. I am also very grateful to my former advisor at Reykjavik University, Dr. Gísli Hjálmtýsson, for introducing me to the world of networking research and for encouraging me to start this PhD quest. Now the destination is near but the journey has been both joyful and rewarding. Gunnar and Gísli, I hope that this is just the beginning of our collaborations.

I would like to thank all the people at LCN for a stimulating and friendly atmosphere. Special thanks go to my long-term Greek office mates, Ilias and Ioannis, with whom I have been sharing fate, both professionally and on a national level, during the last few years. I also thank my former colleagues at Reykjavík University for encouragement, inspiration and lots of fun, in particular Bjöörn Brynjúlfsson and Heimir Póór Sverrisson.

Several people have contributed to the work presented in this thesis and, apart from those already mentioned, I would like to thank my other co-authors: Kristján Valur Jónsson, Dr. Martin May, Dr. Vincent Lenders, Dr. Vladimir Vukadinovic, Dr. Franck Legendre, Sylvia Kouyoumdjieva, Dr. Emre Yavuz and Ljubica Pajevic. I would also like to thank my licentiate opponent Dr. Cecilia Mascolo for valuable comments and discussions on my work.

I thank the organizations and entities that have financed my research: STINT, Rannís and the Graduate School of Telecommunications (GST).

Finally I would like to thank my family for eternal support, love and care. My parents have always been very supportive and encouraging and I know that my late mother would have been very proud of me - I dedicate this thesis to her, Bergljót Guðjónsdóttir. My parents-in-law have also been extraordinary helpful and always been around when needed. Takk Sigga og Skúli! Last but not least I thank my wife Erla and my two sons Emil and Erlingur - you are the meaning of my life.

Ólafur Ragnar Helgason
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contents</td>
<td>v</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td><strong>I Concept and feasibility</strong></td>
<td>7</td>
</tr>
<tr>
<td>2 Background and related work</td>
<td>9</td>
</tr>
<tr>
<td>2.1 Internet content distribution</td>
<td>9</td>
</tr>
<tr>
<td>2.2 Wireless ad hoc networks</td>
<td>11</td>
</tr>
<tr>
<td>2.3 Opportunistic content distribution</td>
<td>13</td>
</tr>
<tr>
<td>2.4 Discussion</td>
<td>16</td>
</tr>
<tr>
<td><strong>3 On the effect of cooperation</strong></td>
<td>17</td>
</tr>
<tr>
<td>3.1 Epidemic modelling</td>
<td>17</td>
</tr>
<tr>
<td>3.2 Epidemic model for content dissemination</td>
<td>18</td>
</tr>
<tr>
<td>3.3 Conclusions</td>
<td>27</td>
</tr>
<tr>
<td><strong>4 A simulation framework for opportunistic networking</strong></td>
<td>29</td>
</tr>
<tr>
<td>4.1 Approach</td>
<td>30</td>
</tr>
<tr>
<td>4.2 Design and implementation</td>
<td>31</td>
</tr>
<tr>
<td>4.3 Summary</td>
<td>35</td>
</tr>
<tr>
<td><strong>5 On the effect of mobility</strong></td>
<td>37</td>
</tr>
<tr>
<td>5.1 Does mobility matter?</td>
<td>37</td>
</tr>
<tr>
<td>5.2 Modelling human mobility</td>
<td>38</td>
</tr>
<tr>
<td>5.3 Simulation configuration and connectivity metrics</td>
<td>40</td>
</tr>
<tr>
<td>5.4 Effect of speed and arrival process</td>
<td>44</td>
</tr>
<tr>
<td>5.5 Effect of scenario</td>
<td>47</td>
</tr>
<tr>
<td>5.6 Effect of node interactions</td>
<td>50</td>
</tr>
<tr>
<td>5.7 A street model</td>
<td>52</td>
</tr>
<tr>
<td>5.8 Conclusion and discussion</td>
<td>63</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

In recent decades communication networks have had a major effect on human society. The Internet has revolutionized how we access, publish and disseminate information and wireless communication has influenced our lifestyle and significantly affected how humans communicate. The Internet and wireless networking are merging more than ever before as computing evolves from being mainly a desktop activity to being an activity that today is commonly carried out by people on the move. This evolution is being expedited by the appearance, and immense popularity, of a new generation of powerful, attractable and affordable handheld devices such as smartphones and mobile tablets.

From hosts to contents

The Internet was originally designed to provide hosts (i.e. end-user computers) with access to other remote hosts. During the Internet’s infancy, computing was mainly performed by few large mainframe computers and the purpose of the Internet was to interconnect these computers and enable users to access their resources through remote terminals. This is evident by some of the first successful application protocols such as telnet and ftp. Telnet allowed for remote terminal access and provided users with command-line access and ftp enabled the uploading and downloading of files to and from a remote computer. Thus the Internet was mainly used for host-to-host communication when accessing resources at a remote machine such as computing power, memory, storage, files, programs and printers.

With the advent of personal computers (PC’s) and the World Wide Web in the early 1990’s, the Internet took off on a large scale and at the same time its usage started to change. With powerful computers commonly available in homes and offices, the need for remote resource access was heavily reduced and the Internet was primarily being used to pull information from web-servers; i.e. the focus started to move from hosts to contents. During the late 1990’s the most popular websites were search engines (e.g. Yahoo and AltaVista), commercial sites such as Amazon and various news sites where contents were mainly generated by a small group of
1. Introduction

Figure 1.1: Global mobile data traffic in PB/month for 2009-2010 and a forecast for 2011-2015 [18].

professionals. Today, this has changed. With the so-called Web 2.0 applications, such as Facebook, Twitter and Youtube, information is no longer only generated by a small group of professionals but it is created by the users themselves and shared with a broad community with matching interests. Today, content is being generated and consumed in massive amounts and thus the Internet has evolved from originally being a network for accessing remote hosts to being a network that today is mainly used for accessing contents. This evolution is stretching the limits of the Internet’s architecture and currently there are various ongoing research efforts towards a content-centric next generation Internet [2, 43, 57].

From wired to wireless

Mobile cellular networks have predominantly been used for carrying voice traffic, but this is changing rapidly. The ‘smart’ new generation of mobile devices are powerful multimedia platforms and due to the popularity of these devices, contents are frequently being produced and accessed by users on the move. As a result of this, mobile data traffic in wireless cellular networks is currently growing at a rapid pace, as shown in Figure 1.1. In March 2010, Ericsson announced that data traffic has surpassed voice traffic in their cellular infrastructure covering all parts of the world and it is predicted that mobile data traffic will double roughly every year [68, 18]. It is foreseen that matching this growth with a corresponding capacity increase in the wireless infrastructure networks will be a significant challenge in the
near future. This evolution calls for new architectures for accessing content on the Internet and how it is disseminated to users and in particular to the mobile users.

This thesis addresses the feasibility, performance and design of content centric architectures for disseminating information between mobile users. In particular, we consider opportunistic networking approaches that can be used as an alternative to infrastructure wireless networks or where infrastructure networks are unavailable.

Opportunistic content distribution

The fundamental idea behind opportunistic content distribution is that users share and exchange contents directly when the wireless devices of two users are within direct communication range. Opportunistic content distribution is therefore enabled by the cooperation of users and wireless radios that can associate in an ad-hoc peer-to-peer manner. Mobility of the users contributes to the dissemination of contents since new contact opportunities occur when people move and interact socially. Transfer opportunities thus typically arise when people with matching interests meet in public transportation, at restaurants, shopping malls or in urban areas in general.

With opportunistic content distribution we envision a seamless content distribution model that decouples the dissemination of information from traditional Internet based platforms while still enabling users to obtain contents while on the move. This is in contrast to the all-or-nothing approach in how information is accessed today: one is either connected to the Internet with unlimited access to information or one is disconnected with very limited access to information that is non-local to the device in hand. One of the goals of opportunistic content distribution is therefore to bridge this dichotomy in connectivity.

Opportunistic content distribution has some characteristics and properties that are very different from current wireless content distribution approaches.

Infrastructure independent. It does not depend on fixed infrastructure or Internet connectivity. It can therefore be used in regions where infrastructure is absent or very limited, such as in remote rural areas or developing regions. It can also be used to maintain communication when infrastructure is disrupted, such as due to natural disasters, warfare or oppression [94].

Favourable scaling. Scaling properties are inherently different from that of infrastructure wireless networks: the system performance improves with the number of participating nodes. Popular contents will inherently have high availability and be replicated across several nodes, suggesting that performance could be good. This suggests that opportunistic networking can co-exist with, and possibly offload, infrastructure systems [99].

Network neutrality. There is a growing concern that governments or network infrastructure providers limit the availability of contents or services to those that they, or their affiliated partners, provide. The decentralized nature of the
1. Introduction

opportunistic content dissemination approach makes it very difficult to impose censorship or restrictions on content, platform or the kind of equipment used. Opportunistic content distribution therefore promotes network neutrality [72].

**Inherent locality.** It is intrinsically based on locality and facilitates simplicity in sharing of data between devices in direct communication range. The opportunistic content distribution mode therefore supports location-based services and the idea of pervasive and ubiquitous computing: a large number of communicating heterogeneous devices permeating and assisting the everyday life of humans.

Opportunistic content dissemination among mobile devices can extend, and co-exist with, wireless infrastructure-based dissemination from the Internet. Applications that use the opportunistic dissemination mode should tolerate a modest amount of delivery delay. This property applies to many content-centric applications that already exist such as blogs, file sharing, podcasting, software updates and more. In addition we believe that this communication mode can spark off new types of applications that take advantage of one or more of the properties listed above.

**Thesis scope and outline**

In this thesis we study the performance and feasibility of opportunistic content distribution in urban areas. In particular, we study how dissemination performance is affected by user mobility and cooperation. Then we introduce our design of an opportunistic content distribution system that is based on the publish/subscribe paradigm [27]. Our system can utilize connections with access points, when in range, and distribute contents opportunistically from mobile node to mobile node otherwise.

It has been recognized that the publish/subscribe paradigm is well suited for content centric networking [2, 92, 48]. One of its main benefits is that it de-couples content publishers and consumers (i.e. subscribers) in both space and time and therefore separates content from location semantics. This separation is what unites the design of some of the proposed future Internet architectures [43, 92, 93]. We believe that in a wireless opportunistic network this separation is crucial for realizing efficient content dissemination. Due to unpredictable mobility, short contact durations and lack of end-to-end connectivity, traditional end-to-end delivery between a client and server is hard to realize. Binding content to a location is therefore not a likely path to success. Subscribers should be able to obtain content from any node that has it and is willing to share. Also, since popular content will inherently have high availability and be replicated across several nodes, performance could be good, assuming that nodes are altruistic and share content with peers.

The main contributions of this thesis are the following.
1. It reports on the feasibility and performance of opportunistic content distribution using both analytic and simulation models.

2. Using a Markovian model inspired by epidemic modelling [22, 3], we study how the willingness and capability of nodes to cooperate affects the spreading of content. Our results show that performance is highly dependent on node cooperation but good performance can still be achieved if cooperation is limited.

3. We have designed and implemented a simulation framework for the OMNeT++ discrete event simulator for simulating opportunistic networking.

4. We study how mobility affects the connectivity of wireless nodes using Legion Studio; a sophisticated agent-based pedestrian mobility simulator. We explore which elements of mobility affect mobile communication and our results indicate that capturing the scenario in which mobility occurs is important while exact modelling of speed distribution, arrival process or node interactions is less important.

5. We present an analytic model that captures the opportunistic spreading of content in an urban area that we model as a grid of streets. The model allows us to study the effect of various system parameters and give insight into how they affect the spreading. We find that content can spread well even at relatively low arrival rates. We compare the analytical results with simulations using Legion Studio with good agreement.

6. System design: We present the architecture and design of PodNet, a content-distribution system based on a publish-subscribe paradigm. PodNet uses a decentralized content solicitation scheme that allows the distribution of content between mobile devices without requiring Internet connectivity and infrastructure support. This scheme is efficient in the presence of intermittent contacts and short contact durations. The system design addresses key issues, in particular the structuring of content to facilitate efficient lookup and matching of contents and a content solicitation protocol that enables discovery and download of contents in the mobile ad-hoc domain. Although our focus is mainly on peer-to-peer content distribution among mobile wireless nodes we also present a design for the Internet that enables seamless content dissemination between the wired Internet domain and the mobile ad-hoc domain. On the Internet, PodNet uses single source multicast to implement scalable and efficient delivery of published content to subscribers over the current unchanged Internet architecture.

7. We have implemented our design for the Android mobile platform and also in the OMNeT++ simulator and we evaluate our system with small scale experiments and large scale simulations.
1. Introduction

8. We have started studying the use of an energy efficient dual radio subsystem for opportunistic networking. With such a system, nodes use a low-power radio for neighbour discovery and the high power radio is only used for the downloading of content items and suspended otherwise. We provide some initial numbers that evaluate the potential energy savings that can be achieved with such a system.

This rest of this thesis is structured as follows. Chapter 2 gives a background overview and discusses related work. In chapter 3 we study how node cooperation affects opportunistic spreading of content. Chapter 4 describes our design and implementation of a simulation framework for opportunistic networking and in chapter 5 we look at the role of user mobility. Chapter 6 presents our design of a mobile middleware for opportunistic content distribution and in chapters 7 and 8 we present the corresponding implementation and system evaluation respectively. Chapter 9 describes our design and evaluation of an energy efficient radio subsystem for opportunistic networking. We conclude in chapter 10 and discuss directions for future work.
Part I

Concept and feasibility
Chapter 2

Background and related work

2.1 Internet content distribution

The Internet was designed according to the end-to-end principle which (in brief) postulates that the operations of a communication protocol should be implemented at the communication end-points whenever possible. This design lead to a sleek network core, specialized for forwarding data, while most of the other functionality is pushed to hosts at the network edge. The traditional Internet client-server model is an offspring of this design and the host-to-host communication service provided by the Internet. The best known instantiation of the client-server model is associated with the world wide web where a web client (usually a browser) accesses contents, such as html documents, images and audio/video files, from a web server. Since the Internet provides host-to-host communication service, each content item needs to be identified with a Uniform Resource Locator (URL) that associates a host with the content. In the traditional client-server model this host is the web server where the content is stored.

For small to medium websites the traditional client-server model usually works well, but the model has a scalability problem when web sites or certain contents at a server become very popular. This can lead to the server becoming overloaded, resulting in slow response or even total unavailability. The problem can, to a small extent, be deferred by adding more hardware at the server but in many cases this is not enough since eventually the network links connecting the server to the core network become overloaded as well and network capacity becomes the new bottleneck.

There are two main approaches that have been used to address this problem for the Internet, content distribution networks (CDN) and peer-to-peer networks. In a CDN, the capacity problem is solved by replicating and caching contents on servers in physically different locations. Thus when a client requests a content item she is redirected to the physically closest server or a server with low load. Usually this redirection is implemented by means of the Domain Name System (DNS).
CDN’s are transparent to the end user and the content provider pays the CDN provider that runs and operates the infrastructure servers (such as Akamai). The CDN provider has to implement the mechanisms needed for content replication and caching across its servers, such as updating contents if they are modified, replacing items in the content cache etc. CDN’s are mainly used by large content providers such as YouTube, Facebook, large news sites, search engines and Olympic web-sites.

Various peer-to-peer systems have been proposed for file sharing and scalable content delivery over the Internet. Napster is generally regarded as the first peer-to-peer file sharing system. Napster users registered their shared contents at a central server. The central servers were thus used for indexing contents and for peer and content discovery while file transfers were performed directly between peers. Napster was blocked by court order in 2001, two years after it started operating, the reason being that the platform was heavily used to exchange copyrighted contents in opposition with copyright holders. Other systems however quickly emerged such as Gnutella, Kazaa and eDonkey. In contrast to Napster, most of these systems used a decentralized approach for content discovery which made it much more difficult to monitor which contents were being exchanged and by whom. From a technical viewpoint it is particularly interesting that the motivating factor towards implementing a fully decentralised approach was not just to achieve scalability. In this case, an equally important and appealing attribute of a fully decentralised file-sharing architecture was that monitoring which contents were being exchanged is much harder than in a centralised system, which significantly complicates a potential lawsuit issue. With the advent of popular legal Internet platforms for obtaining contents (such as YouTube, Itunes, Spotify and last.fm) illegal peer-to-peer sharing of copyrighted contents has diminished. There are however still many systems around that are raising controversy and this is likely to continue in coming years.

BitTorrent [9] is a protocol for file downloading that has become very popular on the Internet. It is mainly used for downloading large files and it can significantly reduce the load on the original publisher of the file as well as distribute the load in the network. With BitTorrent a file is split into a number of pieces and the basic idea is that downloaders concurrently obtain the individual pieces of the file in a disjoint manner from different sources. As soon as a downloader obtains a piece of the file it starts acting as a seeder (i.e. uploader) for the file thus relieving the original source from having to upload it to every peer. The availability of a file and its pieces is maintained by a tracker which the peers periodically communicate and synchronize with. The tracker is usually implemented by means of a centralised server but decentralised implementations using a distributed hash table also exist. BitTorrent is used by some controversial sites for peer-to-peer exchange of audio and video files, most notably the Pirate Bay site. Unlike most of the previously mentioned systems, BitTorrent has however also seen widespread non-controversial use such as for distributing software and software updates and for distributing contents from authorised publishers.

Podcasting is a content distribution method that has gained large popularity on the Internet. It is mainly used for distributing recorded audio or video material
2.2 Wireless ad hoc networks

A wireless ad hoc network is a network where nodes do not assume or rely on the existence of fixed infrastructure such as access points or base stations. Instead, communication takes place over wireless links of nodes that are within direct communication range. Nodes can therefore communicate directly with their neighbours and if they also act as relays and forward data on behalf of others, the communication can be extended to nodes multiple hops away. Ad hoc networks are decentralised, self-configuring networks that can be formed on the fly without central administration when nodes detect the presence of neighbours. Wireless ad hoc networks are commonly classified according to node mobility: in a mesh network the nodes are usually assumed to be stationary while in a mobile ad hoc network (MANET) the nodes can move. In this thesis we consider mobile wireless ad hoc networks.

Research on wireless ad hoc networks dates back to the work on packet radios and the Aloha system in the early 1970’s [1]. From these early days and into the mid or late 1990’s much of the research focused on multiple access control (MAC) and routing. The purpose of multiple access control is to share the wireless channel fairly between the nodes and avoid collisions that can occur when two or more nodes that are within interference range transmit simultaneously. In an infrastructure network, the base station can be used to synchronise nodes and control access to the medium to ensure that collisions do not occur. Ad hoc networks however lack a...
centralised controller and therefore the TDMA and FDMA approaches commonly used in infrastructure systems cannot be used. Instead, some early systems used the ALOHA MAC protocol while different variants of CSMA/CA are commonly used today (such as in the IEEE 802.11 and 802.15.4 standards).

The goal of MANET routing protocols is to establish and maintain end-to-end communication paths between nodes. The general assumption is that the wireless nodes cooperatively participate in constructing routes as well as in forwarding traffic on behalf of other nodes. Many different protocols have been proposed [84] and most of them can be classified as either reactive or proactive (i.e. table-driven) protocols. Reactive protocols construct routes on demand from the source node while proactive protocols try to maintain up-to-date routing information between all nodes in the network. With these MANET routing protocols, nodes can only communicate when there exists a connected path between them. Therefore, MANET routing will only perform well when node density is relatively high so paths can quickly be repaired if a link breaks or when mobility is low such that links break infrequently. For some of the potential applications of MANET’s this is however not necessarily the case.

Opportunistic- and delay tolerant networks (DTN) are a class of mobile ad hoc networks that do not assume continuous and uninterrupted connectivity between a source and destination. In fact, in a DTN a connected path may perhaps never exist. Just like in traditional MANET’s, the goal is to provide networking where devices have heterogeneous capabilities and infrastructure is absent or sporadically deployed. The main difference is that opportunistic- and delay tolerant networks focus on networking in challenged environments where communication links frequently break or are subjected to long delays. Examples are interplanetary and deep space communication [13], networking in battlefield and disaster areas, providing Internet to remote and/or developing areas [78, 65], mobile sensor networks [87] and different systems for mobile peer-to-peer networking between human-carried devices, which is the category of our work.

As with almost all types of communication networks, routing is a topic that has received significant attention in delay tolerant networks. In these types of networks the problem translates to delivering a message from source to destination, in the presence of intermittent connectivity. The general approach for routing messages in such networks follows the store-carry-and-forward paradigm. A source node stores its messages in a buffer and when it meets the destination, or a node which is likely to meet the destination in the near future, the message is forwarded to the neighbour. This way, the mobility of nodes is exploited in an effort to physically carry messages closer to the destination for eventual delivery. Message can thus be carried by intermediate nodes that are neither the source or destination.

Various routing protocols have been proposed for DTN’s and most of them study the trade off between delivery performance and the network resources used for message delivery. On the one hand, the simplest but least efficient routing strategy is to have the source wait until it meets the destination. This results in minimal overhead but potentially suffers from long delays and low delivery probability. On the other hand, if the source and all intermediate nodes replicate the message to all
nodes they meet, delay is minimized at the expense of large message and storage overhead in the network [97]. Therefore routing protocols generally try to limit the number of message replicas while at the same time obtaining low delay. This can be done by limiting the hop-count or total number of messages [88, 44] or by only replicating messages to nodes that have a high probability of meeting the destination. This meeting probability can for example be based on contact history [66, 12, 6] or, when available, mobility schedule or dedicated message ferries [103].

The DTN architecture [16] is a general communication architecture to enable communication in the presence of intermittent connectivity. It consists of an overlay, called the bundle layer, which operates above the transport layer. The architecture specifies the format of variable length application data units, called bundles. The goal is to deliver bundles from a sender to a receiver in the presence of intermittent and opportunistic connectivity, possibly over a wide range of different networks using different transport protocols. This is achieved by assuming that nodes store, carry and forward bundles to cope with link outages. The DTN architecture is node-centric and mainly focuses on unicast delivery of messages although some extensions for group communication have been proposed [24]. It’s design philosophy is therefore significantly different from the content-centric approach we advocate in this thesis and permeates our system design in chapter 6.

2.3 Opportunistic content distribution

The work in this thesis belongs to a particular class of opportunistic and delay tolerant networks: mobile peer-to-peer networking between human-carried devices. The application scenario that we consider is not unicast communication but the distribution of content to a group of receivers. Assuming that nodes cooperate and share content with their peers, popular content will inherently have high availability in the network which suggests that performance could be good for popular contents. Also, by decoupling content from node identities, caching or content replication strategies can be used to further increase content availability for less popular content. Most current smartphone architectures have advanced multimedia capabilities. Since these devices are commonly equipped with radios that can operate in ad-hoc mode, they offer a promising platform for opportunistic networking. We therefore believe that content-centric applications are well suited for the wireless opportunistic environment.

The system design proposed in chapter 6 in this thesis builds on previous work on the PodNet project [81], particularly the work in [53] and [63]. [53] presents the original idea of a delay-tolerant broadcast system and evaluates its feasibility in an urban area while [63] introduces podcasting as an application for delay-tolerant networks. Our design extends and generalizes previous work on the PodNet system by defining a general purpose publish/subscribe system for challenged networks and by specifying a detailed middleware and protocol design.

Recently there have been systems proposed that utilize peer-to-peer contacts of
mobile hosts for distributing and sharing information in a similar way as we consider in our work. 7DS [76] is a system for opportunistic dissemination among mobile devices of Internet data objects, identified by URL’s. As such, 7DS was originally intended mainly for extending web browsing and e-mailing of mobile nodes beyond the reach of access points. There is now ongoing work in updating and extending the original 7DS architecture to provide a generic platform for communication in disruption-tolerant networks [69].

PeopleNet [70] is a distributed geographic database where information is stored at peoples mobile devices. Query requests and responses are forwarded from a mobile device via the cellular network to the geographic location which supports that particular type of request (named Bazaar). Users within the Bazaar then spread queries via peer-to-peer contacts. When a response is found for a query request the user who placed the query request is informed through the cellular infrastructure. In contrast to what we propose, PeopleNet heavily relies on a fixed infrastructure and is targeted at seeking information in contrast to broadcasting information. BlueTorrent [51] is an opportunistic file sharing application for Bluetooth enabled devices. The concept of distributing large files using small atomic chunks follows our approach. However, BlueTorrent relies on Bluetooth whereas our design is not restricted to Bluetooth. Furthermore, we propose to structure the data in the network into feeds and rely on a receiver-driven content dissemination protocol.

Haggle [91] is an architecture for mobile devices that facilitates the separation of application functionality from the underlying network technology. The goal is to allow applications to operate seamlessly across different networking environments and architectures. It achieves this by late just-in-time binding of network interfaces, protocols and names. In the absence of infrastructure connectivity, applications are bound to interfaces that allow nodes to associate at the link level in ad-hoc mode (such as Bluetooth or 802.11). Nodes share data with their peers and metadata is also exported to allow for data searches locally between different applications on the same device as well as between peering nodes. Haggle is thus not a strict protocol architecture for disruptive networks but rather proposes a node design that allows nodes and applications to adapt to the network connectivity level. In a recent redesign of the Haggle architecture [73], the focus is shifted away from point-to-point communication towards a more content-centric view, therefore making it more similar to the interest driven dissemination of the our architecture, originally proposed in [53]. There are however some notable differences in the design. In Haggle, content is unstructured but in both systems, contents have associated meta-data that facilitates searching and organizing of contents. We use a hierarchical content structure to associate content items with particular content feeds. Moreover, the actual data objects are further divided into fixed size chunks to facilitate distributed and disjoint content download. The Haggle architecture is push-based as opposed to the pull-based solicitation protocol in our architecture. It is unclear how the push-based approach avoids redundant data transmissions of already available content and how nodes can prioritize downloads according to their own preferences.
2.3. Opportunistic content distribution

Publish/subscribe and content forwarding

Our design makes the minimal assumption that nodes share the content items that they themselves are privately interested in. Content dissemination is thus purely interest-driven and nodes do not necessarily cache or forward any content beyond what they are privately interested in. The architecture does however not prevent applications from implementing their own caching strategies and we believe that the content structure and solicitation protocol of our design facilitate this. Content replication and forwarding is one of our primary directions for future and on-going work and below we overview recent efforts and methods.

In recent years, publish/subscribe systems have been widely adopted in the context of wired networks and the Internet [27]. These systems generally include an infrastructure broker mechanism that is responsible for delivering relevant publications to interested subscribers. Often this is achieved by some form of multicast but due to the difficulty in maintaining multicast semantics in opportunistic networks (such as group membership and the temporal relations between group membership and message publication), a traditional multicast model does not seem applicable [104].

SocialCast [20] is a content-based routing scheme for publish/subscribe communications in a DTN environment. In SocialCast, a publisher originally delivers a fixed number ($\gamma$) of copies of the published message to carrier nodes. A message carrier will deliver a copy of the message to subscribers it meets or delegate the message to another node that is selected as a more feasible carrier. The carrier selection is based on a comparison of utility values that reflect the probability of a node to be co-located with another node that is interested in the message. SocialCast does not assume that nodes share their private content but that they provide a buffer for storing content that is selected according to the carrier selection algorithm. Contents can therefore only be obtained from one of the $\gamma$ message carriers.

Another proposal is presented in [101] where mobile nodes run a community detection algorithm. In each community, the nodes with the highest closeness centrality (i.e. the shortest path to all other nodes in the community) act as message brokers. Nodes publish a message by delivering it to a broker in their community. Broker nodes in different communities exchange messages and if a broker receives a message that matches an interest of one of its community nodes, the message is flooded among the community.

ContentPlace [10] is a system that exploits social relationships between nodes to decide which content to replicate. Each node advertises the content feeds to which it is subscribed along with the list of available items. When a node meets a peer, it decides which items from the peer should be replicated locally according to a replication policy. The local node calculates the utility of all its local objects and the objects in the peer’s cache and selects from those objects from the peer that maximize the local utility of its cache. The utility values for items are composed from an estimation of the access probability and the availability of the item in
each community. These can be tuned to implement different policies for how nodes populate their buffers (such as whether a node should try to assist the current community that it is in or the most frequently visited community). One drawback with this method is that it requires the estimation of many parameters and that it, just like [101], relies on a community detection mechanism (which in turn assumes the estimation of even more parameters).

The work in [40] shows that the caching problem can be modelled as an optimization problem that maximizes a system welfare objective and that a system optimal assignment of content feeds to users can be found using a centralised greedy approach. Then they propose a distributed allocation algorithm that approximates the centralised optimal assignment. In short, the algorithm works as follows: When two nodes meet, with a certain probability, one of the nodes picks a random cached feed from peer and replaces with one of its own cached feeds. The probability of performing the exchange is based on the fraction of nodes that are subscribed to, or caching the exchanged feed, and in the paper they show how this parameter can be locally estimated by a node. This strategy seems particularly interesting with respect to our system and as part of our future work, we would like to explore whether it can improve dissemination performance while keeping the increase in resource consumption and overhead to a minimum.

2.4 Discussion

Despite years of research there are not many mobile ad hoc systems that have been deployed and many of the devised protocols and mechanisms have not seen practical use. We believe that the end-to-end connectivity approach, adopted from wired networks into traditional MANET’s, is one of the main reasons for lack of success and that the looser and less restrained connectivity paradigm advocated by opportunistic and delay tolerant networks has greater potential to succeed. Embracing mobility as an information carrier and incorporating connectivity disruptions into the system design, as opposed to treating it as an exceptional error state, avoids much of the complexities required for trying to maintain an end-to-end communication path in a mobile environment. At the same time we acknowledge that some types of applications may be difficult to support with opportunistic networking, in particular applications with tight delay constraints such as real-time audio or video conversations. By additionally designing the system for providing users with access to contents instead of hosts we believe that a further simplification can be achieved. Delivering a message to a single particular host in a mobile environment with opportunistic node contacts is a difficult problem. Popular content is however likely to be available, and exist on many different nodes, suggesting that content dissemination may perform well for the common case.
Chapter 3

On the effect of cooperation

In this chapter we study the feasibility and performance of opportunistic content distribution with respect to node cooperation. We introduce stochastic Markov chain models that capture the dynamics of opportunistic content spreading and use these to study the effect of cooperation among the mobile nodes and how limited node resources, such as battery lifetime and confined storage, affect the content distribution process. We indirectly model node mobility through the distribution of the inter-meeting time between nodes, i.e. the elapsed time between successive contacts of the same pair of nodes. This method is commonly used in the field of mathematical epidemiology which studies the spreading of an infectious disease among a population.

3.1 Epidemic modelling

The mathematical field of epidemic modelling has a long history where both stochastic and deterministic models are used to study the spreading of infectious diseases [22, 3]. In this chapter we present stochastic models for studying the effect of cooperation on opportunistic content distribution. The models are adapted and extended from well-known models from the field of epidemic modelling. Epidemic modelling has received considerable attention from the networking research community, as there are many scenarios that arise which are analogous to the spreading of epidemics. Generally these models however need some adaptation to be applied to opportunistic networks. In opportunistic networks we are usually more interested in the asymptotic value of delay until a certain node becomes infected (the unicast routing case [33, 87, 88]), or until all nodes are infected (multicast/broadcast case studied in our work), while the original models study parameters such as the threshold for a major outbreak, the duration of an epidemic, effect of vaccination and the fraction of population ultimately infected.
3. On the effect of cooperation

3.2 Epidemic model for content dissemination

We consider \( N \) nodes moving in a closed area and assume that the nodes are interested in a common content item, i.e. they are all subscribers to the same content feed. Initially at \( t = 0 \), a single node publishes a content item that is of interest to all the other \( N - 1 \) nodes. The content item is assumed to be small such that whenever two nodes meet, they establish a contact that is long enough for them to download the item (the content item might also be divided into small chunks that can be downloaded during short contacts).

At each time instant, a node in the system is in one of three states: infected, susceptible or recovered. An infected node is one carrying the content item of interest and is willing to share it with others. When an infected node meets a susceptible node, the latter obtains the content and becomes infected itself. A node that has obtained the content but is not willing to share it any more is recovered.

We assume that the mobility of the nodes is such that the inter-meeting times between any pair of nodes can be modelled by IID random variables. We further assume that the inter-meeting times are exponentially distributed with rate \( \lambda \). This assumption allows us to model the content distribution process using continuous time Markov chains. Some experiments indicate that inter-meeting times in real life do not follow an exponential distribution [17, 19]. Therefore we compare our results with simulations where the inter-meeting times have other probability distributions.

We identify three basic types of cooperation schemes that differ in the degree of node-cooperation.

**Type I:** No cooperation in spreading the content among the nodes. Nodes can only obtain content directly from the node that publishes it.

**Type II:** Nodes are willing to share private content that they currently have. We study both unlimited and limited sharing.

**Type III:** In addition to sharing content that the nodes are privately interested in, nodes are willing to solicit and carry content that is not of direct interest to themselves.

In the models that follow we study the content distribution process for a single content item that is published by one of the nodes at time \( t = 0 \). For all the cooperation strategies, we look at the performance of the content dissemination time. To characterize this process, we look at two metrics: the overall and the individual delivery times. The **overall delivery time**, denoted by \( T_{\text{o-dt}} \), is the time until content has spread to all the \( N \) nodes. Hence, \( T_{\text{o-dt}} = \max\{T_i\}_{i=1,\ldots,N} \), where \( T_i \) is the time at which node \( i \) obtains the content. The **individual delivery time**, denoted by \( T_{\text{i-dt}} \), is the time that it takes for an arbitrary node to obtain the content. When all the nodes are identical and if the mobility process of the nodes are identical and independent then the sequence of random variables \( \{T_1, T_2, \ldots, T_N\} \) are IID and they have the same distribution as \( T_{\text{i-dt}} \). From a performance viewpoint,
3.2. Epidemic model for content dissemination

\[ T_{odt} \text{ is a measure of the performance of the system as a whole while } T_{idt} \text{ is a measure of the system performance as seen from an arbitrary node.} \]

### 3.2.1 Type I: no sharing

Under this scheme there is no collaboration between the individual nodes in distributing the content item. We assume that the original publisher is the only infected node in the system and that it remains infected forever (this node could for example be an access point). When one of the other \( N - 1 \) susceptible nodes in the system meets the original publisher it obtains the item but immediately becomes recovered. This non-cooperative model does thus not capture the relaying effect of the content spreading process but it serves as a worst-case baseline in performance comparison with the cooperative models.

We denote by the random variable \( X(t) \) the number of infected and recovered subscribers at time \( t \). The process \( \{ X(t); t \geq 0 \} \) is a pure birth process with rate \( \lambda_i = (N - i) \lambda \) for all states \( i = 1, ..., N - 1 \) as shown in Figure 3.1. The continuous time Markov chain (CTMC) for this process is absorbing with \( \{ i = N \} \) as the absorbing state and all other states are transient.

In a CTMC, the state sojourn time is exponentially distributed with rate equal to the sum of the rates going out of the state. Therefore the time to absorption from the initial state \( \{ i = 1 \} \) is equal to the overall delivery time \( T_{odt} \), and its mean is calculated as the sum of the mean time spent in each of the transients states. We therefore have that

\[
E\left[ T_{odt} \right] = \sum_{i=1}^{N-1} \frac{1}{\lambda_i} = \frac{1}{\lambda} \sum_{i=1}^{N-1} \frac{1}{(N - i)} = \frac{1}{\lambda} \sum_{j=1}^{N-1} \frac{1}{j} = \frac{1}{\lambda} H_{N-1} \quad (3.1)
\]

where \( H_n = \sum_{i=1}^{n} \frac{1}{i} \) is the \( n \)-th harmonic number. It is well known that the harmonic series does not converge when \( n \to \infty \). An asymptotic expansion for the harmonic numbers is \( H_n = \gamma + \ln(n) + O\left(\frac{1}{n}\right) \) where \( \gamma \) is Euler’s constant. Thus we have

\[
E\left[ T_{odt} \right] = \frac{1}{\lambda} \left( \gamma + \ln(N - 1) + O\left(\frac{1}{N-1}\right) \right) \quad (3.2)
\]

from which we deduce that \( E\left[ T_{odt} \right] \in O(\ln(N)). \)

The susceptible nodes can only obtain content from the single provider. Since all the nodes nodes are identical and the inter-contact times are IID, the mean
individual delivery time is equal to the mean of the inter-contact time, or
\[ E[T_{idt}] = \frac{1}{\lambda} \quad (3.3) \]

### 3.2.2 Type II: cooperative sharing

With this degree of cooperation, nodes share content that they are privately subscribed to. Whenever an infected node meets a susceptible node, the content item is downloaded and the susceptible node becomes infected. First we consider the case of unlimited sharing where each node remains infected forever. Then we turn to a model where infected nodes can recover and therefore stop sharing contents.

#### Unlimited cooperation

Denote by \( X(t) \) the number of infected nodes at time \( t \). The process \( \{X(t); t \geq 0\} \) is a pure-birth Markov process with positive transition rates

\[
\begin{array}{ccc}
\text{From} & \text{To} & \text{Rate} \\
 i & i+1 & i(N-i)\lambda \\
\end{array}
\quad (3.4)
\]

The Markov chain in (3.4) is a variation of the simple stochastic Markovian epidemic [22] and its transition diagram is shown in Figure 3.2. Similarly to the previous case, \( \{i = N\} \) is the only absorbing state and we are interested again in the mean overall delivery time, \( E[T_{odt}] \), which equals the mean time to absorption from state \( \{i = 1\} \), as well as the mean individual delivery time \( E[T_{idt}] \).

Let \( R_{i,i+1} \) denote the time that it takes for the process, starting from state \( i \), to reach state \( i+1, i \geq 1 \). \( R_{i,i+1} \) is exponential with rate \( i(N-i)\lambda \) and thus

\[ E[R_{i,i+1}] = \frac{1}{\lambda} = \frac{1}{i(N-i)\lambda} \]

The time it takes all the nodes to obtain the content is \( T_c = R_{1,N} \) and its expected value is \( E[R_{1,N}] = E[R_{1,2}] + E[R_{2,3}] + \cdots + E[R_{N-1,N}] \). We thus have

\[ E[T_{odt}] = E[R_{1,N}] = \sum_{i=1}^{N-1} E[R_{i,i+1}] = \frac{1}{\lambda} \sum_{i=1}^{N-1} \frac{1}{i(N-i)} \quad (3.5) \]
and since
\[ \sum_{i=1}^{N-1} \frac{1}{i(N-i)} = \frac{1}{N} \left( \sum_{i=1}^{N-1} \frac{1}{i} + \sum_{i=1}^{N-1} \frac{1}{N-i} \right) = \frac{2}{N} H_{N-1} \] (3.6)
we have the following asymptotic expansion for \( E[T_{odt}] \)
\[ E[T_{odt}] = \frac{2}{\lambda N} \left( \gamma + \ln(N - 1) + O \left( \frac{1}{N-1} \right) \right) \] (3.7)
where \( \gamma \) is Euler’s constant. Since \( \lim_{x \to \infty} \frac{k_1 \ln(x-1)}{x} = 0 \) and \( \lim_{x \to \infty} \frac{k_2}{x} = 0 \), where \( k_1 \) and \( k_2 \) are constants, we have that \( E[T_{odt}] \in O(1) \).

Thus when the nodes utilize the peer contacts to cooperatively share content the mean overall delivery time is bounded by a constant and does not grow to infinity when \( N \to \infty \) as is the case when there is no peer cooperation (3.2).

To obtain \( E[T_{idt}] \) we denote by the random variable \( T_{k,N-1} \), the time until \( k \) out of the \( N-1 \) susceptible nodes have become infected. We also introduce the event \( B = \{ A \text{ given susceptible node is the } k\text{-th to become infected} \} \). Then we have that
\[ E[T_{idt}] = \sum_{k=1}^{N-1} E[T_{k,N-1}] P\{B\} = \sum_{k=1}^{N-1} E[T_{k,N-1}] \frac{1}{N-1} \] (3.8)
where \( P\{B\} = \frac{1}{N-1} \) is uniform since all nodes are identical and inter-contact times are IID. \( E[T_{k,N-1}] \) is the mean time that it takes the Markov chain (3.4) to reach state \( \{ i = k+1 \} \) and therefore we have
\[ E[T_{k,N-1}] = \frac{1}{\lambda} \sum_{i=1}^{k} \frac{1}{i(N-i)} \] (3.9)
and thus
\[ E[T_{idt}] = \frac{1}{\lambda(N-1)} \sum_{k=1}^{N-1} \sum_{i=1}^{k} \frac{1}{i(N-i)} \] (3.10)
By writing out terms in the double sum, it is easily shown that
\[ \sum_{k=1}^{N-1} \sum_{i=1}^{k} \frac{1}{i(N-i)} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{N-2} + \frac{1}{N-1} = H_{N-1} \]
3. On the effect of cooperation

The mean individual delivery time is thus

\[ E[T_{idt}] = \frac{1}{\lambda(N-1)} H_{N-1} \]  

(3.11)

from which we can deduce that \( E[T_{idt}] \in O(1) \).

Limited cooperation

The model above assumes that nodes are willing to cooperate in an unlimited manner, i.e. once the content item is obtained the node never deletes or stops sharing it, and remains infected forever. Since the power resources of a mobile node are generally limited nodes may want to limit their sharing in order to save power. We can model this behaviour by assuming that nodes stop sharing the item after a certain time or after it has been shared \( k \) times.

**Time limited sharing:** Here we assume that when a node becomes infected with the content item, it shares it for an exponentially distributed time with rate \( \mu \). After this time has passed, it stops sharing and becomes recovered. Denote by \( X(t) \) the number of infected plus recovered nodes at time \( t \) and \( Y(t) \) the number of infected nodes at time \( t \). We have the initial conditions \( X(0) = 1 \) and \( Y(0) = 1 \).

The process \( \{(X(t), Y(t)); t \geq 0\} \) is a two dimensional Markov chain and its positive state transition rates

\[
\begin{array}{ccc}
\text{From} & \text{To} & \text{Rate} \\
(i, j) & (i+1, j+1) & j(N-i)\lambda \\
(i, j) & (j-1) & (j-1)\mu \\
\end{array}
\]  

(3.12)

This Markov chain is absorbing and the absorbing states are \( \{(X(t), Y(t)) = (N, j); \forall j\} \). We point out that we do not allow the content to vanish from the area since we assume that \( 1 \leq Y(t) \leq X(t) \) for all \( t \). In other words, we assume that the node that brought the content initially remains infected forever.

**Download limited sharing:** Here we restrict the uploading such that nodes will share each content only \( k \) times. Once a node has reached this limit, it recovers and stops sharing. Let \( X(t) \) be the number of infected and recovered nodes at time \( t \) and \( Y_m(t) \) the number of infected nodes who have to share the content \( m \) times before becoming recovered, where \( m = 1,...,k \). Thus right after a susceptible node obtains the content it will become infected and since it has never shared its content \( Y_1 \) will increase by one. Also, after an infected node that has shared its content \( n \) times \((n = 0,...,k-1)\) meets a susceptible node, \( Y_{k-n} \) will decrease by one and \( Y_{k-n-1} \) will increase by one.

When \( k = 1 \) each node will only share the content once and only a single node will be infected at each time until the content is fully spread. In fact, when a susceptible node meets an infected node, they will become recovered and infected respectively and thus there is always just one infected node at a time in the system. Therefore this stochastic process is the same as the one-dimensional Markov chain.
3.2. Epidemic model for content dissemination

\{X(t); t \geq 0\} for the No cooperation strategy in Figure 3.1. It should also be clear that when \( k \to \infty \), a node will share a content infinitely many times and then this scenario is the same as the previous unlimited strategy in 3.4.

For a general \( k > 1 \) the Markov process for the system will consist of \( k+1 \) random variables since we have to count the number of nodes for each of the sequence of variables \( \{Y_m(t); t \geq 0\}_{m=1,...,k} \) as well as the total number of nodes that have been infected. Here we give the model for the case \( k = 2 \). Initial conditions are \( X(0) = Y_2(0) = 1, Y_1(0) = 0 \). The positive state transition rates for the process \( \{(X(t),Y_1(t),Y_2(t)); t \geq 0\} \) are

\[
\begin{array}{ccc}
\text{From} & \text{To} & \text{Rate} \\
(i,j,k) & (i+1,j+1,k) & k(N-i)\lambda \\
 & (i+1,j-1,k+1) & j(N-i)\lambda \\
\end{array}
\] (3.13)

The Markov chains for the limited sharing cases are two and \( k \)-dimensional respectively and they do not easily lend themselves to closed form solutions. Therefore we resort to numerical analysis and simulations.

3.2.3 Type III: generous sharing

In this section we model the scheme where node cooperation is generous in the sense that nodes are willing to assist in distributing other contents than those that they are privately interested in. Thus, in addition to sharing the private content a node is interested in, it is willing to solicit and cache public content for the benefit of others.

As before, we assume that there are \( N \) nodes in the area privately subscribed to a channel, herein referred to as subscribers. Assume moreover that there are \( M \) other nodes in the area which are willing to assist in spreading the content. Since these \( M \) nodes do not have private interest in the content we refer to them as assistants. We model this system as a CTMC where the following events cause state transitions

(a) Infected subscriber meets susceptible subscriber

(b) Infected subscriber meets susceptible assistant

(c) Infected assistant meets susceptible subscriber

Note that assistants never infect each other, they can only be infected in a contact with an infected subscriber. This is similar to two-hop multicopy forwarding [33, 34]. More aggressive spreading such as unrestricted multicopy forwarding [33] would presumably result in faster spreading but at the expense of requiring more resources from the participating nodes.

In the following model we assume unlimited sharing for both subscribers and assistants. We denote by \( X(t) \) the number of infected subscribers and by \( Y(t) \) the number of infected assistants at time \( t \). At time \( t = 0 \) a single node publishes a new
content item, giving initial conditions $X(0) = 1$, $Y(0) = 0$. The positive transition rates for the process $\{(X(t),Y(t)); t \geq 0\}$ are

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Rate</th>
<th>Event</th>
<th>(3.14)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(i,j)$</td>
<td>$(i+1,j)$</td>
<td>$i(N-i)\lambda + j(N-i)\lambda$</td>
<td>(a) and (c)</td>
<td></td>
</tr>
<tr>
<td>$(i,j+1)$</td>
<td>$(i,M-j)\lambda$</td>
<td></td>
<td>(b)</td>
<td></td>
</tr>
</tbody>
</table>

When $M = 0$ this model is equal to the type II cooperation with infinite storage in (3.4).

### 3.2.4 Numerical results

All the CTMCs in the previous subsections reach an absorbing state when the number of infected plus possibly recovered nodes reaches $N$, i.e. when all nodes have obtained the content. We obtain closed-form expressions for the mean overall delivery time and the mean individual delivery time for the first two cooperation strategies. For the other cases we resort to numerical methods. Our methodology for numerically calculating the mean time to absorption is described in appendix A.

In [33] the authors show that for the random waypoint and random direction mobility models the inter-contact rate $\lambda$ is approximated by

$$\lambda \approx \frac{2\omega r E[V^*]}{L^2}$$  \hfill (3.15)

where $\omega$ is constant specific to the mobility model ($\omega \approx 1.3683$ for the random waypoint model). If we assume that the average relative speed between two nodes is $E[V^*] = 1.5$ m/s, that the area is a square with sides of length $L = 100$ meters and that the communication range of the nodes is $r = 10$ meters, we obtain $\lambda = 0.0041$ s$^{-1}$ which corresponds to an average inter-meeting time of $1/\lambda = 244$ s or roughly 4 minutes. These parameters could for example represent a square in a city where pedestrians are carrying a device equipped with a short-range radio such as Bluetooth.

In Figure 3.3a we plot the mean overall delivery time $E[T_{odt}]$ as a function of the node population $N$. We compare the type 1 (no sharing) and type 2 (cooperative sharing) cases with unlimited and limited sharing. We clearly see that when nodes cooperate, the delivery time is significantly reduced. From the figure we can observe the asymptotic behaviour of the type 1 and type 2 cooperation models that was derived in equations (3.2) and (3.7). The figure also shows the effect of limiting sharing in type 2 cooperation. For the time-limited sharing case we have plotted for $\mu = \lambda, 10\lambda$ and $100\lambda$ and we see that when $\mu \to \infty$ the mean sharing time tends to zero and the system tends to one of type I with no cooperation between the nodes. When $\mu \to 0$ the system tends to a type II system with unlimited sharing. The results for the download-limited case are particularly interesting. If each node only shares the content twice the overall delivery time is only slightly
longer than what it is in the unlimited cooperation case. Moreover, it seems to have the same asymptotic behaviour as the unlimited cooperation case (although we have not proved this analytically). This simplifies buffer management since content can be safely deleted after it has been shared \( k \) times and it suggests that nodes can conserve energy by limiting the number of uploads, and hence the number of transmissions, while only slightly reducing performance. One remark is however that reducing the number of transmissions does not necessarily result in significant energy savings. Some radios, most notably 802.11, have a very high idle energy cost. In other words, energy consumption in idle mode is not negligible compared to the energy consumption when transmitting and receiving. In chapter 9 we address this issue further and propose an energy efficient radio architecture that has the goal of reducing the idle cost associated with 802.11.

In the download-limited sharing model of (3.13), the state space grows quite fast as the number of nodes increases. The size of the transition-rate matrix \( Q \) of the Markov Chain grows as \( O(N^3) \) and therefore we can only solve (3.13) exactly for small systems (a maximum of \( N = 36 \) in our evaluations). However, even with systems of this size the performance differences between the individual cooperation schemes are very evident. For larger \( N > 36 \) we resort to simulation of (3.13). For each value of \( N \), a mean value of \( T_{odt} \) is obtained from 1000 simulation runs (95% confidence intervals are omitted from figure since they cannot be distinguished from the mean value).

Figure 3.3b shows the mean individual delivery time \( E[T_{idt}] \) for the different cooperation models. Data for the download-limited type 2 case is obtained from simulations and, as for the overall delivery time, it is interesting to see that individual delivery time is only slightly longer for the limited cooperation than what it is
3. On the effect of cooperation

Figure 3.4: (a) $E[T_{odt}]$ for log-normal (simulated) and exponential inter-meeting time distributions for type I and type II cooperation. (b) Comparison of $E[T_{odt}]$ for type II and Type III cooperation with different number of assistants ($M$).

for the unlimited case. It is also interesting to see that initially, when $N$ goes from 1 to approximately 20, there is a sharp decrease in $E[T_{odt}]$ which then levels out as $N$ increases. This suggests that for obtaining good performance, an initial mass of nodes is needed and a further increase beyond that will only slightly improve performance.

In Figure 3.4a we study the effect of the Markovian assumption in our analytic models. It has recently been proven that the inter-meeting time between two independent mobile nodes decays at least exponentially fast for commonly used mobility models, such as the Random Waypoint and Random Direction mobility models, as long as the area boundary is finite [14]. However, some experiments indicate that the inter-contact time distribution has a heavy tail and obeys a power-law [17] while other results are more optimistic with the log-normal distribution being shown to be a good match for three commonly used reference sets of traces [19].

Figure 3.4a therefore compares results from the Markovian analytic model with results obtained by simulating the type 1 no cooperation and type 2 unlimited sharing models with a log-normal inter-meeting time distribution with the same mean and variance ($1/\lambda$ and $1/\lambda^2$ respectively). Simulation results from 1000 runs are plotted with 95% confidence intervals for the type I case but these are omitted for the type II case since they can barely be distinguished from the mean value in the plot. Direct numerical comparison shows some quantitative difference. In both cases the log-normal distribution results in a longer delivery delay. Despite these numerical differences we conclude that, from a qualitative viewpoint, the assumption of exponentially distributed inter-meeting times in our Markov models captures the behaviour of the system, in particular when comparing the difference
3.3. Conclusions

Figure 3.4b shows the effect of assisting nodes on the content distribution process. We assume that sharing is unlimited both for subscribers and assistants (see equation 3.14). Clearly, the assisting nodes have the largest effect when there are few subscribers. When the system is already large, assisting nodes have only marginal effect. This suggests that for promoting fairness and giving new content a chance to spread, assistants should also help in spreading less popular content. A similar result was found, based on simulations only, in [63] but here we give an argument based on an analytical model.

3.3 Conclusions

In this chapter we have addressed the issue of cooperation in opportunistic content dissemination. We have identified three basic types of cooperation where the degree of node collaboration and generosity differs. For each of the cooperation types identified we give a Markov chain model that captures the dynamics of content spreading. We have also addressed and captured in our models how limited resources at the mobile nodes can affect the content spreading when nodes limit their sharing. In particular, we study the effects of limited content lifetime at the nodes and limited number of uploads from a node which may be used to reduce the power consumption required for data transmissions.

The main conclusions from our models and numerical results are the following.

- Performance of the content distribution is highly dependent on the degree of cooperation of nodes. When there is high collaboration (type II and type III) the content spreading time is much shorter than when cooperation is limited or none (type I). We have seen that the mean content distribution time grows with the number of nodes \( N \) as \( \mathcal{O}(\ln(N)) \) when nodes are non-cooperative. When nodes cooperate by sharing content, the mean content distribution time is bounded by a constant and is \( \mathcal{O}(1) \).

- Limiting the number of times a node (or assistant) shares a content item gives only a slightly worse performance than when nodes have infinite storage.

- For obtaining good performance, a critical mass of nodes is required for spreading the content. After reaching this critical mass, a further increase will not significantly contribute to the delivery time performance.

- A similar effect is seen when looking at the effect of assisting nodes. Assistants are most helpful for small systems (less-popular content) while for popular content with many subscribers their effect is marginal. This suggests that for promoting fairness in distributing channels and giving new channels a chance to spread, assisting nodes should also help the spreading of less popular channels.
Chapter 4

A simulation framework for opportunistic networking

Performing experimentation on mobile wireless networks is a difficult task. It is non-trivial to capture meaningful measurement results because of the number of external factors influencing the measurements. Reproducing the environment between individual experiments is almost always impossible because of interference, fading, mobility patterns, randomness in the MAC layer contention, weather etc. Moreover, performing experimentation with a large number of mobile nodes is expensive and difficult to manage. These are some of the main reasons for why researchers and developers turn to simulation in evaluating protocols and mechanisms for wireless mobile networks.

OMNeT++ [98] is a public-source simulation platform that has primarily been used for simulating communication networks. Some of the main benefits of OMNeT++ are its modular design and strong GUI support. The MiXiM [67] and inetmanet [42] frameworks for OMNeT++ provide extensions to the core simulator for supporting mobile wireless network simulations. However, they lack support for some of the features that are characteristic for the class of opportunistic- and delay-tolerant networks.

In opportunistic networks, the intermittent connectivity of nodes arises because of node mobility and delivering messages is non-trivial because nodes carrying them may leave the area under consideration. Simulating networks of this kind needs, in many cases, to be done with an open system approach. The current frameworks however have very limited support for simulating open systems, dynamic arrivals and departures of nodes. They also have limited support for importing mobility and contact traces from external measurements or other simulators. The modules we have implemented for our mechanisms make use of some of the functionality provided by the OMNeT++ MiXiM framework and can therefore be seen as extensions to it.
4. A Simulation Framework for Opportunistic Networking

4.1 Approach

We propose simulations which are driven by tracefiles, and our mechanisms allow for simulation of open systems where mobile nodes can dynamically arrive and depart from a simulation scenario. Our design supports two separate approaches for simulating opportunistic networking, a *mobility driven* and a *contact driven* approach. With the mobility driven approach, mobility patterns of the nodes are specified in a mobility tracefile and during a simulation, node contacts arise when two or more nodes are within communication range. With the contact driven approach, the time of node contacts and their durations are specified in a contact tracefile. This approach does therefore not directly simulate the mobility of nodes but contact events, which are a consequence of node mobility, are used to drive the simulation.

A common design for these approaches is that we separate mobility generation of nodes from the core protocol simulations. There are various benefits associated with this. First, it facilitates importing both synthetic and real mobility and contact traces. In particular, it allows for importing traces from external mobility or contact generators and mobility patterns from GPS tracking methods or real contact traces. Second, it allows mobility patterns to be generated in more flexible high-level programming environments. Also, by having a single mobility module that handles traces, instead of a special module for each type of mobility, the simulator core code can be simplified. Third, with mobility traces it becomes easier to re-execute the same mobility scenario on different protocol parameters and it facilitates running same mobility scenarios on different simulators. Properties of the mobility process can also be analyzed more easily offline. Finally, a standardized trace file format for the mobile network simulation community would increase the inter-operability between different tools for generating mobility patterns and performing simulations.

Our contributions to the OMNeT++ community are threefold:

- We present XML formats for node mobility and contact traces. The structure of the XML files is specified by XML-schemas.
- We provide OMNeT++ modules for dynamically creating and destroying nodes during a simulation run and for implementing our mobility and contact-driven approaches.
- We have implemented a set of tools for generating node mobility patterns and for converting output from external mobility generators to our XML trace format.

Our code is available at [https://github.com/olafur/mixim](https://github.com/olafur/mixim) and [http://code.google.com/p/opponet](http://code.google.com/p/opponet)\(^1\).

---

\(^1\)An older version, backwards compatible with previous versions of OMNeT++ and MiXiM
4.2 Design and implementation

Our design allows highly dynamic mobile simulation scenarios to be created, where nodes can enter and exit the scenario during the course of a simulation run. Our mechanisms are implemented in OMNeT++ modules that adhere to the modularized design of the simulator. The key components are the NodeFactory, TraceMobility and ContactNotifier modules. The NodeFactory dynamically manages nodes in the simulation, using scripted events in a tracefile, which specifies the create and destroy times of individual nodes, as well as mobility waypoint updates or contact events.

A simulation node in our framework is a compound module of OMNeT++ modules. When using the mobility driven approach, one of the node sub-modules is a mobility module of type TraceMobility (which is derived from the BaseMobility module of MiXiM). For a contact driven simulation, a ContactNotifier module manages establishment of contacts and contact break events as specified in a contact tracefile.

Figure 4.1a shows an OMNeT++ simulation example scenario with a single global factory object of type NodeFactory. The scenario contains a single stationary gateway node and several dynamically created mobile hosts. The ChannelControl from MiXiM manages communication over the wireless channel. Figure 4.1b shows a typical mobile node object from the example. A generic mobility block, named navigator in this case, can take on the role of any BasicMobility-derived object at creation time, including that of TraceMobility. The Blackboard is utilized for notifying sub-modules within the node of location updates. The other modules making up the node are a wireless interface card, a protocol module and controller, all of which are specific to the particular example scenario (see [49] for a complete description of the scenario).

4.2.1 Tracefile format

Two distinct types of tracefiles are supported: Mobility traces define creation times and positions for each dynamically created node in the scenario. Additionally, destroy events and waypoint updates may be associated with each created node. Contact traces define a set of contact establishment and break events for a population of nodes. Both the mobility and contact tracefiles are in XML format, which imposes strict syntax and semantics on the file structure and allows parsing with open-source software libraries and tools. The structure of the tracefiles is defined by XML schemas, allowing tracefiles to be validated using commonly available schema validators. We now describe the structure of the tracefiles in more detail.

Mobility trace

A mobility trace includes a series of node create, waypoint and destroy events to specify the lifetime and mobility of a population of nodes.
Create events specify the arrival of a node with a given id into the scenario at a given time and location. A node type designation enables nodes of various types and capabilities to be instantiated. The type can be used to refer to a corresponding module type for a host defined in the simulator. Upon a create event, the simulator thus instantiates a new object of the particular type. A simulation could for example include pedestrians, vehicles and access points, all differing in capabilities and resources. Other runtime parameters of the node being created can also be specified such as the mobility model, a name to specify a string for display in a GUI environment or output traces. Similarly, an icon can be specified for easier visualization of different node roles or classes.

Waypoint events specify a waypoint change, i.e. the next destination on a nodes journey. A node moves with a fixed speed on the leg between the current and destination waypoints and when the destination is reached the node stops (if no other waypoint is scheduled immediately). Waypoint events are only required when nodes change direction or velocity. Relatively compact tracefiles are thus feasible, even with a large number of generated nodes.

Destroy events specify the departure of a node with a given id from the scenario at a given time. The node is destroyed regardless of any remaining waypoint events.

Listing 4.1 shows an example of a simple mobility tracefile. A single node of type SimpleNode is created at $t = 0.0$ at $(x, y) = (0.0, 0.0)$ and starts moving towards $(x, y) = (10.0, 10.0)$ at $t = 10.0$ s with $v = 2.0$ m/s. It then remains stationary at the final destination until it is destroyed at $t = 60.0$ s.
4.2. Design and implementation

Listing 4.1: A simple mobility tracefile.

```xml
<mobility-trace>
  <create>
    <time>0.0</time>
    <nodeid>1</nodeid>
    <name>Host 1</name>
    <mobilityModel>TraceMobility</mobilityModel>
    <location>
      <xpos>0.0</xpos>
      <ypos>0.0</ypos>
    </location>
  </create>
  <waypoint>
    <nodeid>1</nodeid>
    <time>10.0</time>
    <destination>
      <xpos>10.0</xpos>
      <ypos>10.0</ypos>
    </destination>
    <speed>2.0</speed>
  </waypoint>
  <destroy>
    <time>60.0</time>
    <nodeid>1</nodeid>
  </destroy>
</mobility-trace>
```

Contact trace

In many simulation scenarios the detailed mobility of the nodes is not needed and it is sufficient to instead consider only individual contact opportunities. A contact trace specifies the beginning and end of each contact opportunity in a particular scenario and it can also contain information regarding each contact such as the mean bitrate during the contact, mean signal to interference ratio or contact setup time.

A contact trace is less general than a mobility trace since the contact events are defined by the node mobility and communication range of the wireless radio. Using a contact trace therefore inherently imposes some simplifying assumptions on the simulation since it assumes that each contact can be described as an on/off process with a fixed bitrate. In many simulations, a detailed model of the physical and MAC layers is however not needed. In particular, traces of this kind have recently been generated in opportunistic networking experiments, e.g. [17, 25], and they are commonly used in simulations.

As a bare minimum, contact trace consists of sequences of contact and break events:

- contact events signify a discovery event by a node, specified by a nodeid, of a peer with id peerid at a given time with a given mean bitrate.
- break events signify a broken or lost contact between a node, specified by a nodeid, and a peer with id peerid at a given time.
4. A simulation framework for opportunistic networking

Listing 4.2: A simple contact tracefile.

Listing 4.2 shows a simple example of a contact trace. Node 1 discovers its peer Node 2 at \( t = 5.0 \) s. The contact is broken at \( t = 10.0 \) s.

4.2.2 NodeFactory

The NodeFactory module dynamically creates and destroys nodes during the course of a simulation. A tracefile is parsed at the initiation of the simulation, and an event is created on the simulator event queue for each create and destroy event. The waypoint or contact events are also read at initialization and stored in an associated container, implemented as a STL map of lists, keyed by node id. No such events are however scheduled by the factory; the created nodes are initialized with this information upon creation and each node is then locally responsible for scheduling its own mobility/contact events.

When a scheduled create event is executed during a simulation run, a node of the specified type is instantiated using the appropriate OMNeT++ methods for dynamic node creation. If the node uses TraceMobility as its mobility module, it is initialized with the previously parsed waypoint update information. Similarly, a node employing a ContactNotifier module is initialized with the associated contact and break events at startup. Each node is henceforth autonomous in the sense that it is locally responsible for managing its own mobility or contacts.

When a destroy event occurs during a simulation run, the event handling routine of the NodeFactory is called. It invokes the appropriate OMNeT++ node deletion methods. Note that the NodeFactory can be utilized to instantiate and destroy any type of node, regardless of its mobility model. A trace consisting solely of create and destroy events could thus be used to manage a population of stationary nodes or nodes using some other mobility model that does not use a mobility trace. The NodeFactory is implemented as an OMNeT++ simple module, and resides at the scenario level as shown in Figure 4.1a.

Dynamically creating and destroying nodes on demand admittedly adds somewhat to the overhead of the simulation, since dynamic memory allocation can be a computationally expensive operation. However, this approach is in our opinion

\[
\begin{align*}
\text{Listing 4.2: A simple contact tracefile.}
\end{align*}
\]
superior to the current practice in simulation of dynamic scenarios, where all nodes are created at startup and "flown" into the active region on demand and not destroyed until the simulation stops. This places great demands on the simulator, both in terms of memory and computing power, and it significantly limits simulation scenarios with a large number of nodes. Another related performance issue with our current implementation is that the full simulation tracefile is read and parsed at startup. For large-scale simulations the tracefile can become somewhat big and thus the stored create, destroy and waypoint/contact events can add unduly to the memory requirements. We intend to solve this issue in our future work such that large simulation trace files are read in chunks as the simulation progresses.

4.2.3 TraceMobility module

The TraceMobility module manages the mobility of a node according to a mobility tracefile. It is based on, and extends, MiXiM by deriving from the BaseMobility class. A TraceMobility module can thus be used to enable trace controlled motion in any simulations which currently use MiXiM for OMNeT++. When the NodeFactory creates a new node with a TraceMobility module, it passes the newly created node the list of waypoint events belonging to the node. The TraceMobility module manages the node mobility autonomously thereafter. A periodic event scheduled locally by the module triggers location updates, and the mobile node thus moves in a number of small steps to its next waypoint. After each step, the position of the node is updated and subscriber modules are notified through the Blackboard module of MiXiM. The granularity of the motion is controlled by the length of the update period, which is a configurable simulation parameter.

4.2.4 ContactNotifier module

The ContactNotifier manages the contact events of a node when performing contact-driven simulations. There is no actual mobility of nodes at the simulation level when employing the contact-driven approach. Therefore, nodes in a contact-driven simulation do not have a mobility module, but instead employ the ContactNotifier to publish contact events to subscribing modules of the the node through the Blackboard. Although not a mobility module as such, the ContactNotifier serves a comparable purpose of notifying sub-modules of connectivity status change. ContactNotifier can thus be viewed as replacing the TraceMobility module in the navigator role when using the contact driven approach to opportunistic networking simulation.

4.3 Summary

We provide support for trace-driven simulations in OMNeT++ where mobility is separated from the core protocol simulations. This approach facilitates importing synthetic and real data from external mobility generators, real mobility tracking
4. A simulation framework for opportunistic networking

data and real contact traces. We have described our design and implementation of mechanisms for conducting simulations driven by mobility or contact traces. Our extensions to OMNeT++ and the MiXiM framework consist of the specification of mobility and contact traces, a module for dynamically creating and destroying nodes during the course of a simulation and modules that implement node mobility or node contacts from tracefiles. Our simulation framework will be used extensively for evaluating opportunistic networking scenarios in the remainder of this thesis.
Chapter 5

On the effect of mobility

It is known that mobility significantly affects the performance of wireless communication systems [34, 5, 17]. In opportunistic networking, mobility is an integral part of the system as it contributes to the data dissemination when nodes store, carry and forward messages. In this chapter we study how mobility affects connectivity in wireless communication. Our focus is on common modes of mobile communication, namely where mobile devices are carried by humans in built environments. Our approach is mainly based on a detailed and realistic micro-level simulation model for pedestrian mobility. We use this model to try to quantify how elements such as speed, arrival process, node interactions and scenario affect connectivity. Then we focus more specifically on opportunistic content distribution and present an analytic mobility model for a street scenario. We show how this model can be extended to capture content spreading in larger urban areas and compare the analytic results with the simulation model.

5.1 Does mobility matter?

Many performance evaluations of mobile wireless systems use synthetic mobility models, where nodes move randomly in a closed area. Because of the randomness and infinite sojourn time, nodes in such models might not represent human mobility patterns. Models have been presented to address the shortcomings [100, 45, 71, 26, 60]. Despite the steps taken towards more realistic modelling of human mobility, there are still a number of issues that are left out of the modelling scope. For example, models usually assume a free flow of nodes and do not consider node-to-node interactions. The space where mobility occurs is either not considered at all, or only in a very limited way. Cultural aspects, such as differences in personal space requirements and walking speed, are usually neglected. Moreover, most models are closed systems, and the effect of arrival processes and sojourn times in the observed area are not considered. Studies based on random mobility, without consideration of physical interaction, and without arrival and departures of nodes might give
misleading conclusions on the performance of systems.

Measurements of mobility might provide a means to build realistic mobility models and the measurement traces may be used for simulation studies. Most measurement studies employ coarse temporal and spatial sampling of node positions and therefore only capture human mobility at time-scales of tens of seconds, minutes or even hours [17, 25]. Hence the measurements capture only snapshots of nodal movements, and not the exact paths. Moreover, the under-sampling leads to missed node contacts which in turn might affect the system performance.

We feel that there is still a lack of understanding of which elements of mobility are important when it comes to evaluating wireless communication systems, in particular with respect to the sensitivity of system performance to individual mobility parameters. In this chapter we study how mobility affects the performance of wireless communication systems. Our approach is based on a detailed and realistic micro-level mobility model implemented in Legion Studio; a commercial pedestrian simulator package commonly used by architects and civil engineers for designing and dimensioning large public spaces. The mobility in Legion Studio is based on advanced analytical models [52] that have been calibrated by measurement studies [8]. Our work seeks answers to the following questions:

- How do micro-level mobility parameters (such as speed, arrival process, and personal space requirements) affect the connectivity parameters (such as link duration, contact rate, and path duration) of a mobile wireless system?
- How does the scenario in which mobility occurs affect the connectivity parameters?
- Is connectivity highly sensitive to even small changes in input mobility parameters?
- Are some input parameters more important than others? If so, which?

Furthermore, we address whether mobility can be captured by relatively simple analytic or simulation models to give meaningful performance results, or if other more advanced models are needed. In other words, what are acceptable simplifications and abstractions when modelling mobility for the sake of evaluating wireless systems. For instance, is a rough estimate of input parameters sufficient for performance evaluation or are accurate estimates of the empirical distributions needed?

5.2 Modelling human mobility

Human mobility can be classified in three mobility levels: strategic, tactical and operational mobility [39]. The strategic level describes daily movement patterns of individuals, for example going to work or shopping. Based on the strategic decisions, the tactical level is focused on choosing the travel path (which can be the shortest path for getting to certain destination) depending on the environmental
5.2. Modelling human mobility

factors, e.g. obstacles or congestion. In the end, the operational level depicts the physical process of human movement, focusing on walking speed, physical size of the nodes and interaction with others due to queuing or to avoid traffic.

Each of these levels is likely to affect the performance of wireless communication networks differently. On one side, the decisions taken on strategic and tactical level can affect the inter-meeting times between specific nodes, which is of high importance for delay-tolerant networks. On the other side, operational level decisions are likely to affect node connectivity, as well as durations of different contacts both between nodes that are in each other’s communication range, as well as for nodes which communicate over one (or more) relay nodes. This determines the amount of data that can be transferred over a contact which is of great importance for forwarding contents between nodes or for disseminating data for content distribution applications.

Recently, a line of works has focused on human mobility at the strategic and tactical levels. Some of these works analyse and characterize the mobility effects due to temporal clustering of nodes that belong to the same social network [71, 61] or diurnal regularity in human movements [26, 31]. Unlike these works, our work focuses on micro-level mobility at the operational level and how individual contacts are affected by mobility details. Modelling mobility at this level for the sake of evaluating wireless communication has received less attention but [45] and [4] are two exceptions. The mobility model in [45] captures movements of nodes where obstacles block or restrict their freedom to roam or their signal propagation. It is however quite restrictive in the types of obstacles that can be modelled and it does not capture node-to-node interactions. The work in [4] is perhaps the most closely related work to our study. They find that mobility impacts the connectivity graph of nodes and that different mobility models impact the connectivity graph in different ways. By simulating various ad-hoc routing protocols under the different mobility models they confirm that the choice of mobility model matters and that performance ranking of protocols may vary with the mobility model used. In particular, they find that there is a strong correlation between the link duration and path duration metrics of the connectivity graph [35] and performance of ad-hoc routing protocols. Their mobility models however do not have the same operational-level details as the mobility model we use and they only consider closed systems with relatively few nodes while we study open systems and collect statistics from a large sample.

The importance of correct capturing of pedestrian and crowd mobility has received a lot of interest in the last decade. A number of analytical models have been developed, most of them in the area of social science [36] and urban planning [7, 62], as well as gaming [77]. Some of these models are primarily used for designing and dimensioning large-scale public spaces in order to optimize the flow of large crowds as well as for drawing emergency and evacuation strategies. It is thus of primary importance to realistically capture the structure of the physical space, how nodes interact with it and how they interact with each other. These models therefore better capture micro-mobility of pedestrians than mobility models used in mobile networking. To the best of our knowledge, those studies have until now been kept
5. On the effect of mobility

separately from the field of wireless communications.

For capturing human mobility, we use Legion Studio [62]: a commercial simulation software package initially developed for designing large-scale spaces via simulation of pedestrian behaviours and used mainly by architects and civil engineers. Legion Studio allows importing of AutoCAD drawings of real life structures such as subway stations or grids of streets in urban areas. The multi-agent pedestrian model is based on advanced analytical and empirical models [52] which have been calibrated by measurement studies [8]. In the simulation, each pedestrian is represented by a two-dimensional circular entity which size approximates the size of an actual pedestrian. The navigation decisions of each entity are mostly based on the interaction with other nodes, as well as the interaction with the surrounding environment, although overriding certain choices is possible. Like in real life, the movement patterns follow the least effort principle where each entity tries to minimize its dissatisfaction before choosing its next move. Legion Studio provides different aspects of the dissatisfaction factor: inconvenience, frustration and spatial discomfort. Moreover, Legion Studio incorporates all the three levels of human mobility (strategic, tactical and operational) for each entity in the simulation, thus allowing correct capturing of events like queuing behind slower nodes or at bottlenecks. To resemble reality even more, Legion Studio allows the use of open systems, where entities can enter and leave the system according to a predefined pattern. The ability to record both the spatial and temporal position of each node during its lifetime in the system provides extensive information that can be used for evaluating mobile communication systems based on thorough examination of user behaviours. One of the main contributions of our work is thus the introduction of agent-based simulation, and in particular Legion Studio, for evaluating mobility with respect to communication network performance.

5.3 Simulation configuration and connectivity metrics

Each simulation run conducted in Legion Studio results in a mobility trace file, where the position of each node is captured every 0.6 s. For obtaining the connectivity metrics from the mobility traces we feed the traces into our the OMNeT++ simulator using our extensions described in chapter 4.

We assume that the mobile nodes can communicate over a short-range radio (e.g. Bluetooth or 802.11) and the radio model we employ is simple; if two nodes are within a fixed range $r$ of each other they can communicate. Unless otherwise noted we assume that $r = 10$ m. Physical and link-layer issues such as interference, shadowing, fading or MAC-layer contention are not considered. We realize that these factors affect the performance of wireless systems and that the interplay between them and mobility can be complex. Moreover, the effect of these factors can depend strongly on the particular radio being used. Here we however aim at examining the properties of an ideal mobile wireless system isolating the effect of mobility.
5.3. Simulation configuration and connectivity metrics

Figure 5.1: A simulation scenario: Östermalm area in central Stockholm.

Figure 5.2: A simulation scenario: Two-level subway station.
5.3.1 Mobility scenarios

Our evaluation mainly considers the following two scenarios: an outdoor urban scenario, modelling the Östermalm area of central Stockholm (Figure 5.1), and an indoor scenario, recreating a two-level subway station (Figure 5.2).

The urban outdoor scenario consists of a grid of interconnected streets with a length varying between 20 m and 200 m. Each street has a width of 2 m which is representative for a pavement. The observed area is connected to the outside world with 12 passages, and we assume that nodes enter the area through each of those passages with equal arrival rates denoted by $\lambda$. Upon arrival at an intersection, nodes continue moving along the same street with probability 0.5 or change their direction and turn in an adjoining street with equal probabilities. The active area of the outdoor scenario is 5872 m$^2$. The scenario can be characterized as a high mobility scenario, since nodes move constantly throughout their lifetime in the observed areas. We note that we have experimented with different street selection probabilities and replaced the center streets with a wide square and found that this does not significantly affect our results.

The indoor scenario defines a train platform connected via escalators to the entry-level. Nodes can arrive on foot from any of five entry points of the subway station, or when a train arrives at the platforms. The train arrivals contribute to the burstiness of the node arrivals and departures. Nodes congregate while waiting for a train to arrive at one of the platforms, or while taking a break in the store or the coffee shop at the entry level. Since the station is relatively densely populated, and its structure severely constrains mobility, the physical interaction of nodes is high and leads to queuing. The main bottleneck, where queuing is observed, are the escalators. The active area of the scenario is 1921 m$^2$.

5.3.2 Performance metrics

The main focus of our evaluation is to explore how mobility affects the wireless connectivity graph. Various studies have shown that for mobile wireless systems there is a strong correlation between metrics defined on the connectivity graph and the protocol or system level metrics [5, 17, 4]. The connectivity graph at time $t$ is an undirected graph $G(t) = (V(t), E(t))$ where $V(t)$ is the set of mobile nodes and the edge set $E(t)$ consists of the wireless radio links between nodes. Thus we have that link $(i, j) \in E(t)$ if $D(i, j) \leq r$ at time $t$ where $D(i, j)$ is the euclidean distance between nodes $i$ and $j$. Our evaluation studies the following metrics.

- **Link duration:** This is the time when two nodes are physically within direct communication range. More formally, if there exists an $\epsilon > 0$ such that $(i, j) \in E(\tau)$ for all $\tau \in [t, t + d]$ and $(i, j) \notin E(t - \epsilon)$ and $(i, j) \notin E(t + d + \epsilon)$, then $d$ is defined as the link duration.

- **Contact rate:** We define the contact rate as the number of non-zero link durations per node, divided by the lifetime of the node in the simulation.
5.3. Simulation configuration and connectivity metrics

Since we study open systems, the contact rate is a more suitable metric than the number of contacts, since the rate is (by definition) normalized by the simulation sojourn time of a node.

- **Inter-contact time**: We define the inter-contact time as the elapsed time from the beginning of one contact to the beginning of the next.

- **Path duration**: A path \( P = \{n_1, n_2, ..., n_k\} \) exists between nodes \( n_1 \) and \( n_k \) at time \( t \) if \( \{(n_1, n_2), (n_2, n_3), ..., (n_{k-1}, n_k)\} \subset E(t) \). The path duration is the time during which two remote nodes are physically connected via the same set of relay nodes. It is defined in a similar manner as the link duration. If there exists an \( \epsilon > 0 \) such that \( \{(n_1, n_2), (n_2, n_3), ..., (n_{k-1}, n_k)\} \subset E(\tau) \) for all \( \tau \in [t, t + d] \) and \( \{(n_1, n_2), (n_2, n_3), ..., (n_{k-1}, n_k)\} \not\subset E(t - \epsilon) \) and \( \{(n_1, n_2), (n_2, n_3), ..., (n_{k-1}, n_k)\} \not\subset E(t + d + \epsilon) \) then we define \( d \) as the duration of path \( P \).

The above metrics are important for the performance of most mobile wireless systems. Our definition of link duration includes the node discovery and contact setup time, which can differ from one technology to another. The amount of data that can be transferred over a contact thus depends on the channel bit-rate and the remaining link duration after connection setup. The contact rate is a measure of the contact opportunities available to nodes and the contact rate between any pair of nodes has been shown to be a metric that highly affects the performance of routing protocols in delay tolerant networks [17]. For multi-hop networks the path duration is an important metric that affects the feasibility of ad-hoc routing protocols [5]. The path duration is a measure of the time that two nodes can communicate over a particular path.

5.3.3 Measuring performance metrics

In our evaluation we study both mean values and full distributions of the observed performance metrics where the distributions are estimated using statistical analysis of simulation data. The time resolution of the mobility traces exported from Legion Studio is 0.6 s and therefore we update the connectivity graph with the same resolution and update the measured metrics. The beginning/ending of a contact is assumed to be uniformly distributed in the sampling interval before/after the contact is first/last observed in our simulations. During a simulation run we compute the connectivity graph in every round and update the measured metrics. In all simulations we start collecting data after a warm-up period, i.e. the system starts empty and after the warm-up period we collect statistics from 1000 nodes.

We estimate the path duration metric with the duration of the shortest path (i.e. the path with minimum number of links) since calculating all possible paths is not feasible. Finding the shortest path is usually the goal of ad-hoc routing protocols and therefore it serves as a good approximation. When a path is found between two previously unconnected nodes the path is stored. In the next round
5. **On the effect of mobility**

Figure 5.3: The impact of speed distribution (a) and arrival process (b) on mean link duration for the Östermalm scenario.

...the duration of those paths that still exist is incremented. If a new shorter path is found we keep track of this new path as well as all previous paths between the node pair that still exist.

### 5.4 Effect of speed and arrival process

We use the Östermalm scenario to examine how the input speed distribution and arrival process affect the connectivity metrics and their distributions. Legion Studio allows us to configure the target speed of nodes as an input parameter. Due to queuing behind slow nodes or speeding up when overtaking others, nodes tend to change their actual speed during the simulation runs. Legion Studio allows one to also configure different (pre-defined) cultural profiles that affect node parameters such as level of dissatisfaction, physical size and desired inter-node distance. We have examined those parameters, and we found that the connectivity metrics are insensitive towards them. Therefore we will not discuss them further.

Figure 5.3 shows the effect of target speed distribution and arrival process on the mean link duration. All mean values in this paper are plotted showing 95% confidence intervals. In Figure 5.3a we study three different speed distributions, all with the same mean speed of 1.3 m/s. (i) constant speed, (ii) uniform with minimum speed 0.6 m/s and maximum of 2.0 m/s and (iii) truncated normal with minimum speed 0.6 m/s and maximum of 2.0 m/s. The reason for choosing 1.3 m/s as a mean speed is the study in [75]. Moreover, we conducted experiments with both higher and lower mean speed values, and observed only a shift by a constant factor in the connectivity curves, but no performance changes. In order to facilitate
5.4. Effect of speed and arrival process

In Figure 5.3b we show the effect of different arrival processes on the mean link duration. We have chosen the processes such that they have the same mean \( \lambda \): (i) a Poisson process with rate \( \lambda \), (ii) a 4-stage Erlang where each stage has rate \( 4\lambda \), and (iii) a two-phase hyper-exponential inter-arrival time distribution with arrival rates \( 0.35\lambda \) and \( 5.7\lambda \) in the first, respectively second phase, and selection probabilities 0.31 and 0.69. The variation coefficient of the Erlang and the hyper-exponential distribution is 1/2 and 2 respectively. The speed distribution is truncated normal \((0.6;2.0)\) with mean 1.3 m/s. At lower rates the hyper-exponential arrival process results in longer link durations. We believe the reason for such behaviour is the bursty nature of the hyper-exponential process. Due to the burstiness, nodes entering the scenario have higher chances to be connected upon arrival and stay connected as they move along in a group. Such burstiness is often observed in urban life where nodal clustering is usually caused by traffic lights, train arrivals and departures, etc. The clustering of nodes in the scenario leads to queuing, forcing faster nodes to slow down behind those with leisurely pace, thus increasing the link durations with the surrounding nodes.

Figure 5.4 shows that increasing the transmission range only changes the scale of the link durations but otherwise the results are the same.

In addition to studying the mean values we have also looked at how the distri-
5. On the effect of mobility

The distribution of the link duration is affected by the speed and arrivals. Figure 5.5 shows the histograms of link durations for the three speed distributions under a Poisson arrival process with a per-entry arrival rate of $\lambda = 0.15 \text{ s}^{-1}$. To facilitate a comparison between the histograms they all have equal fixed-width bins and each data point in the plot is the centre of a bin. All the histograms are characterized by a sharp peak in the link durations at approximately 8 s. This is due to contacts with nodes moving in opposite directions. The mean duration of this type of contacts is given by $d = 2r/|v_a - v_b|$ where $r$ is the communication range and $|v_a - v_b|$ is the relative speed of the nodes. For $r = 10 \text{ m}$ and a mean speed of 1.3 m/s the average duration of these contacts is 7.7 s. The figure illustrates how speed affects both the height and the width of the spike. The basic shape of the empirical contact distribution however remains the same in all cases and this shape is determined by the mobility scenario. The histograms for the Erlang and hyper-exponential arrivals have high resemblance and are therefore omitted from the plot.

Figure 5.6 shows the impact of speed and arrival processes on the mean contact rate. The contact rate increases linearly at low arrival rates but at higher rate the node density increases, the linearity is violated, and contact rate grows slower. This is due to the node interactions. We also see that the contact rate is highly insensitive to both arrival process and speed distribution of nodes. We have also found that the same applies to the empirical distribution of the inter-contact time, i.e. it is highly insensitive to both the target speed distribution and node arrival process.
5.5 Effect of scenario

As previously mentioned, the Östermalm scenario is characterized by high mobility. In the Subway scenario, however, nodes can take a break in the coffee shop, or cluster while waiting at the platforms for a train arrival. In Figure 5.7a we investigate how the link duration histogram is affected by the difference in the scenarios. In order to make the comparison easier we select a configuration of the Östermalm scenario that gives comparable node density ($\rho$) with the subway scenario. If we define $\bar{N}$ to be the mean number of nodes in the scenario, and $A$ to be the effective size of the area (i.e. the area in which nodes can move), we can calculate $\rho = \bar{N}/A$. The mean number of nodes $\bar{N}$ can be obtained from Little’s law as $\bar{N} = \lambda_{\text{tot}} \bar{T}$ where $\lambda_{\text{tot}}$ is the mean total arrival rate, and $\bar{T}$ is the mean sojourn time of nodes in the scenario. We measured the sojourn time from our simulations, and use this value for calculating the node density. Thus, we have found that the mean density for the Subway scenario is 0.09 nodes/m$^2$ and that a per-entry arrival rate of $\lambda = 0.15$ s$^{-1}$ gives approximately the same node density for the Östermalm scenario.

When comparing the link duration histograms for the two scenarios we see some significant difference. The histogram for the Subway scenario is more complex with at least two visible peaks, one small peak with very short contacts and another one at about 8 s. It also has a thicker tail due to clustering and waiting, and this is reflected in a significantly longer mean link duration: 22.8 s for the Subway scenario and 13.0 s for the Östermalm scenario.

We use the Kolmogorov-Smirnov (K-S) statistical test to compare the empirical link duration distribution with standard probability distributions: exponential, gamma, weibull, log-normal, rayleigh and pareto. Parametrization of the reference
5. On the effect of mobility

Figure 5.7: Link duration (a) and inter-contact time (b) comparison between the Östermalm and Subway scenarios.

distributions is based on maximum likelihood estimation and a table with the results is given in appendix B. None of the reference probability distributions give a particularly good fit for the link duration but the log-normal distribution gives the lowest K-S statistics (which indicates a better fit) for both scenarios as seen in table B.1. Figure 5.7a however shows that the log-normal fit underestimates the spike at 8 s in the Östermalm scenario and that the log-normal fit does not capture the multimodality of the link duration in the Subway scenario. This suggests that the link duration cannot be accurately modelled by a single standard distribution but should rather be approximated with a mixture of distributions.

We have also studied the distribution of the inter-contact times in the two scenarios and found that they can be well approximated with a Weibull distribution (see Figure 5.7b and table B.1). Recent experiments using Bluetooth devices to log contact statistics of humans have suggested that both link durations and inter-contact times (often referred to as inter-any-contact times in those works) are approximated by power-law distributions [41]. This contradicts our results and in [106] we show that the coarse neighbour discovery process of Bluetooth filters out most of the short contact opportunities and much of the effects of operational level mobility that we consider in this work. Due to the long neighbour discovery time, Bluetooth is not well suited for mobile systems.

Identifying the underlying empirical distributions for the inter-contact time and link duration is important for understanding whether complex mobility models like Legion Studio are necessary to evaluate communication systems or whether some simplifications can be made. If, for a given scenario, the sequence of link durations and inter-contact times of a node are uncorrelated, they could be modelled as a renewal process where the time between contacts and the duration of each
5.5. Effect of scenario

contact are sampled independently from known probability distributions. Figure 5.8 shows
the scatter plots of inter-contact times and link durations for the Östermalm (Figure 5.8a)
and Subway (Figure 5.8b) scenarios. The figures also show the Pearson correlation
coefficient ($r$) and the Spearman rank correlation coefficient ($\rho$) for the
data sets. The Pearson correlation coefficient is a measure of the linear dependence
between two random variables and Spearman’s coefficient is a measure of
whether an increase or decrease in one of the random variables results in an increase
or decrease in the other. We find that both of the coefficients are close to zero which
indicates that there is a low correlation between the inter-contact time and the link
duration. This suggests that the connectivity process can be modelled as a renewal
process. For the scenarios we study in this work we have identified approximations
of the empirical distributions for the inter-contact time and the link duration and
an important direction for future work is to explore if the distributions that match
our scenarios can be generalized to capture other different scenarios. It is also
important to explore further how the parameters for the inter-contact time and
link duration distributions can be estimated from the input mobility parameters
(i.e. node speed, arrival process, scenario size etc.).

It has previously been shown that there is a strong correlation between path
duration at the connectivity graph level and at the routing protocol level [4]. More-
over, the path duration at the connectivity graph level is a good approximation of
the duration of a path as seen by an ad-hoc routing protocol.

Figures 5.9 and 5.10 show the path duration histograms for the Östermalm and
Subway scenarios for paths with 2, 3 and 5 links and for communication range of
10 m and 30 m. We see that there is a significant difference in the path duration
between the scenarios, particularly with a 10 m range. For the Östermalm scenario,
the mean duration of a two-hop path is only 1.7 s while it is 4.3 s for the Subway scenario. A longer communication range leads to longer path durations and for a 30 m range the mean duration of a two-hop path is 5.1 s and 4.7 s for the Östermalm and Subway scenarios respectively. In general, paths are short-lived and therefore it is unlikely that ad-hoc routing will perform well, particularly not in the high mobility Östermalm scenario.

It has been proposed that the path duration distribution can be approximated with an exponential distribution [4, 35]. Tables B.1 and B.2 however show that a log-normal distribution generally gives the best fit among the reference distributions we use. A gamma distribution also gives a good fit in the scenarios we study except for the Subway scenario with a 10 m range where the log-normal distribution gives a clearly better fit. An exponential distribution only gives a comparable goodness of fit for a 30 m communication range when compared to the log-normal and gamma distributions.

5.6 Effect of node interactions

Two of the major strengths of the Legion Studio mobility simulator are accurate modelling of the physical environment and accurate modelling of node dynamics and interactions. The previous subsection studied how mobility is affected by the scenario but in this section we study the effect of node interactions.

When evaluating mobile networks of human carried devices, it is commonly assumed that nodes flow freely and that their physical movements are not restricted by other nodes. Moreover, it is also often assumed that all the nodes in the area
5.6. Effect of node interactions

The graphs illustrate the path duration histograms for the Östermalm (a) and Subway (b) scenarios for a 30 m communication range.

Under inspection are participating in the mobile network. In reality we know that this is however not the case. In dense urban areas, people are constrained by the movement of other pedestrians regardless of whether they are parts of the mobile network or not. To study the effects of node interactions we consider a simple scenario that allows us to focus only on this aspect of mobility. The scenario we study is a 100 m long and 2 m wide street segment with pedestrians arriving at both endpoints. Nodes arrive according to a Poisson process with rate \( \lambda \) and they traverse the whole length of the street with a target speed, drawn from a uniform distribution \((0.6;2.0)\) m/s. For evaluating the effect of node interactions we examine the following configurations:

- No node interaction. Each node maintains its target speed throughout the street and is not affected by other nodes.

- Small-scale node interaction. We model the street segment in Legion Studio and the target speed of nodes is therefore affected by others. In particular, nodes may slow down due to congestion or a slower node in front of it.

- Node interaction with background traffic. We define two types of mobile users in Legion Studio: active nodes that carry mobile devices and can participate in the mobile network, and background nodes who are moving along the street, but do not participate in the mobile network. We denote the arrival rate of background nodes by \( \lambda_{bg} \).

In our evaluation we set the arrival rate of active nodes at each street endpoint to \( \lambda = 0.15 \) s\(^{-1}\) and the Legion simulations are conducted with a background
5. On the effect of mobility

The effect of mobility on wireless communication is significant. Figure 5.11 illustrates the link duration histograms for nodes moving in a street with different node interactions. The arrival rate $\lambda_{bg}$ of 0, 0.1 and 0.2 s$^{-1}$. Our results suggest that the inter-contact time distribution is not sensitive to the node interactions but the link duration histograms show a mild sensitivity to node interactions as seen in Figure 5.11. When node density increases, nodes are forced to slow down due to the interactions with others, which again leads to a slight increase in link duration. For most wireless systems, longer link durations are correlated with better performance. Assuming a free-flow of nodes would therefore lead to a pessimistic evaluation of performance.

5.7 A street model

In the previous sections we have studied how operational level mobility affects wireless communication. So far, we have considered how the connectivity graph is affected without considering a particular application. Now we turn the focus towards opportunistic content distribution and study content spreading among mobile pedestrians in an urban area. First we describe our analytic mobility model for nodes moving in a street of finite length and show how the notion of content can be incorporated into the model to capture the opportunistic spreading of a content item among the nodes in the street. Then we extend use the street model as a building block for modelling mobility and content dissemination in the the Östermalm topology used in the previous sections. The analytical model in this section is based on the mobile infostation model in [102]. We extend it by considering the boundary effects for finite street segments and generalize it to a grid of streets and
5.7. A street model

Figure 5.12: A node with communication range $r$ moves in a street of length $l$. The speed of the node is given by the random variable $V$.

the spreading of content.

**Assumptions**

We assume that pedestrians arrive at the two endpoints of a street of length $l$, according to a Poisson process with arrival rates $\lambda$ and $\lambda'$, as shown in Figure 5.12. When arriving to the street, each node selects a speed $V$ from a uniform distribution, and maintains this speed throughout the street. We denote the minimum and maximum speed with $v_a$ and $v_b$ respectively. As before, nodes have a fixed communication range $r$ and when two nodes are within direct range they establish a radio contact. We assume that, when there is more than one node within range, multiple simultaneous contacts can be established with all neighbours. The random variable $D$ denotes the physical contact duration, which equals the elapsed time that two nodes are physically within direct communication range. In reality, neighbour discovery and contact setup time affects the contact duration and therefore the effective contact duration is generally shorter than $D$. Therefore we are particularly interested in the tail distribution $P\{D > d_{\text{min}}\}$, where $d_{\text{min}}$ represents the minimum useful physical contact duration which may incorporate neighbour discovery, contact setup time and the transmission of a content item.

**Observer node contact rate**

An observer node moving at a speed $v_0$ ($v_0 > 0$) in the street can make three types of contacts:

1. When it overtakes slow nodes with speed $V < v_0$.
2. When itself is overtaken by a faster node with speed $V > v_0$.
3. Contacts with nodes moving in the opposite direction.

We will now show that the arrival of nodes into the observers communication range constitutes a Poisson process with a mean rate $\gamma = \gamma_s + \gamma_f + \gamma_c$, where $\gamma_s$, $\gamma_f$, and $\gamma_c$ are the mean arrival rates of slow, fast and counter-directed nodes respectively. Therefore the observer node can be modelled as an M/G/$\infty$ queue where the service time is the physical contact duration. The following derivation is for the type 2 contacts above. The other cases can be derived in the same manner.
Due to street boundary effects, the observer contact rate depends on the position of the observer node in the street (i.e. it is time-dependent). The contact rate can be affected by node arrivals when the observer is within distance $r$ from the street endpoints. The street therefore consists of three segments and within each of these segments the contact rate is constant.

When the observer node is in the initial street segment, with its position $x \in [0, r]$, all nodes that arrive into the street at $x = 0$ will be directly within the observer’s communication range. Within this segment, the number of contacts with fast nodes therefore follows a Poisson process with mean rate

$$\gamma_f(v_0, t) = \lambda (1 - F_V(v_0))$$

$$= \frac{\lambda}{v_b - v_a} (v_b - v_0) \quad (5.1)$$

In the other two street segments, a contact with a fast node can only occur when a faster node catches up with the observer from behind. If the observer node arrives at time $s_0$ it will leave the street at $s_0 + t_0$, where $t_0 = l/v_0$ is its sojourn time in the street. A later node arriving at time $s_1$ ($s_1 > s_0$) will result in a type 2 contact with the observer node if

$$s_1 + T < s_0 + t_0 \quad (5.2)$$

where $T$ is the street sojourn time of the arriving node. Since $T = l/V$, the distributions of $T$ and $V$ (denoted by $F_T(t)$ and $F_V(v)$) are related through

$$F_T(t) = P\{T \leq t\} = P\{V > l/t\} = 1 - F_V(l/t)$$

$F_V(v)$ is the cumulative distribution function of a uniform distribution with minimum speed $v_a$ and maximum speed $v_b$ and thus we have that

$$F_T(t) = \begin{cases} 0 & t \leq l/v_b \\ \frac{v_b - t}{v_b - v_a} & l/v_b < t < l/v_a \\ 1 & t \geq l/v_a \end{cases} \quad (5.3)$$

Therefore we obtain from (5.2) and (5.3) the probability that the observer makes a contact with a faster node

$$p_f(s_1) = P\{s_1 + T < s_0 + t_0\} = F_T(s_0 + t_0 - s_1)$$

Since nodes enter the street according to a Poisson process, the total number of type 2 contact events that occur are Poisson with mean rate

$$\lambda \int_0^\infty p_f(s_1) \, ds_1$$
5.7. A street model

\[
\begin{align*}
&\gamma_f(v_0, t) = \frac{\lambda}{v_b - v_a} (v_b - v_0) \\
&\gamma_s(v_0, t) = \frac{\lambda}{v_b - v_a} v_0 \ln \frac{v_a}{v_f} \\
&\gamma_c(v_0, t) = \frac{\lambda}{v_b - v_a} (v_b - v_a + v_0) \ln \frac{v_a}{v_f}
\end{align*}
\]

Table 5.1: Observer contact rate for fast, slow and counter-directed nodes (\(\leftarrow\) means "same as in previous column").

In summary, the number of contacts for the observer node follows a non-homogeneous Poisson process. The full time-dependent rate of contacts with slow, fast and counter directed nodes is given in table 5.1.

Contact duration

We will now give a sketch of the derivation of the distribution of the contact duration, denoted by \(F_D(d) = P\{D \leq d\}\). As in the previous section, we derive the duration from the viewpoint of an observer node with speed \(v_0\). The conditional distribution \(F_{D|V=v_0}(d)\) is the weighted sum of the distributions for each type of contacts

\[
F_{D|V=v_0}(d) = \frac{1}{\sum_{i \in \{s,f,c\}} \gamma_i} (\gamma_s F_{D_s}(d) + \gamma_f F_{D_f}(d) + \gamma_c F_{D_c}(d))
\]
where $D_s$, $D_f$ and $D_c$ are the contact durations for slow, fast and counter-directed nodes.

We now outline the case when the observer node makes a contact with a faster node (i.e. $V > v_0$). We have that

$$D_f = \frac{2r}{V - v_0}$$

and therefore

$$F_{D_f}(d) = P\{D_f \leq d\} = P\left\{V > \frac{2r}{d} + v_0\right\} = 1 - F_{V|V>v_0}\left(\frac{2r}{d} + v_0\right)$$

where we now have to condition the speed distribution on $V > v_0$. The conditional probability density $f_{V|V>v_0}(v)$ is

$$f_{V|V>v_0}(v) = \frac{f_V(v)}{P\{V > v_0\}} = \frac{1}{v_b - v_0}$$

when $v > v_0$

which is the density of a uniform distribution with support $v \in [v_0, v_b]$. Therefore we obtain

$$F_{D_f}(d) = \begin{cases} \frac{v_b - v_0 - 2r/d}{v_b - v_0} & \text{if } d \leq \frac{2r}{v_b - v_0} \\ \frac{2r}{v_b - v_0} & \text{if } d > \frac{2r}{v_b - v_0} \end{cases}$$

(5.6)

By deriving $D_s$ and $D_c$ in a similar manner, we obtain an expression for $F_{D|V=v_0}(d)$ in equation (5.5). The contact duration distribution $F_D(d)$ for an arbitrary node can then be found by averaging $F_{D|V=v_0}(d)$ over $V \sim \text{uniform}(v_a, v_b)$. For simplicity and ease of presentation, the derivation above assumes that the street is long enough so that contacts are not affected by the street boundaries. Near the street boundaries, contact durations can be affected when a node arrives and other peers are within a distance $r$ from the street entry. Similarly, nodes can exit the street in the middle of a contact near the street end. Incorporating node boundaries complicates the mathematical handwork but otherwise it is conceptually the same. Our full model considers these effects due to street boundaries and they are incorporated in all results in the rest of this chapter.

**Content dissemination**

So far we have not captured the notion of content in our model but focused on obtaining the content duration distribution from the street model. Now we assume that a fraction of the nodes arriving to the street are infected with a content item that is of interest to the other susceptible nodes. If a contact involving an infected
and a susceptible node is longer than a minimum value \(d_{\text{min}}\), the content will be downloaded by the susceptible node, which becomes infected. In our model, the minimum contact duration thus models the contact setup time plus the time it takes to download the content item.

We denote by \(p_0\) and \(q_l\) the fraction of infected node arrivals from the street endpoints. We denote by \(p(x)\) the probability that a random observer node, moving in the forward direction, has obtained the content at position \(x\) in the street. The probability that the observer possesses the content at \(x + \Delta x\) is

\[
p(x + \Delta x) = p(x) + (1 - p(x))\theta(x) \tag{5.7}
\]

where \(\theta(x)\) is the probability that it meets and connects to an infected node in \([x, x + \Delta x]\).

Since each contact type constitutes a Poisson process, the probability that the observer makes at least one contact with a faster node in \(\Delta t = \Delta x/v_0\) is

\[
1 - e^{-\gamma_f \Delta t}
\]

Only contacts with infected nodes with a duration longer than \(d_{\text{min}}\) will result in the observer node becoming infected. The probability that a faster node is infected is \(p(x)\) and the probability that the contact duration is longer than \(d_{\text{min}}\) is \(\bar{F}_{D_f}(d_{\text{min}}) = 1 - F_{D_f}(d_{\text{min}})\). The contact rate of infected faster nodes, whose contact duration is longer than \(d_{\text{min}}\), is thus \(\gamma_f \bar{F}_{D_f}(d_{\text{min}}) p(x)\) and the probability of observer infection from a fast node in \([x, x + \Delta x]\) is therefore

\[
1 - e^{-\gamma_f \bar{F}_{D_f}(d_{\text{min}}) p(x) \Delta x} \tag{5.8}
\]

where \(\gamma_s \bar{F}_{D_s}(d_{\text{min}}) p(x) + \gamma_c \bar{F}_{D_c}(d_{\text{min}}) q(x)\) is the probability that a node arriving from the other street entry is infected at \(x\).

Since \(1 - e^{-\varepsilon} \approx \varepsilon\) for \(\varepsilon \ll 1\), equation (5.8) can be simplified. Further, by averaging over the observer speed, which is uniformly distributed on \([v_a, v_b]\), the un-conditioned \(\theta(x)\) can be written as

\[
\theta(x) = (a p(x) + b q(x)) \Delta x \tag{5.9}
\]

where \(a\) and \(b\) are constants.

By substituting (5.9) in (5.7), letting \(\Delta x \to 0\), and assuming for simplicity that \(\lambda = \lambda'\), we obtain the set of differential equations

\[
\begin{align*}
p'(x) &= (1 - p(x))(a p(x) + b q(x)) \\
n'(x) &= -(1 - q(x))(b p(x) + a q(x))
\end{align*}
\]

which can be solved with the initial conditions \(p(0) = p_0\) and \(q(l) = q_l\).
5. On the effect of mobility

5.7.1 Performance results

Our analytic model has some simplifying assumptions in order to obtain a tractable model. In particular, it assumes a free flow of nodes where node mobility is not affected by other pedestrians. Therefore we compare the analytic results with the Legion street model in section 5.6.

In Figure 5.13a we have plotted the complementary distribution of the contact duration, $P\{D > d_{\text{min}}\} = 1 - F_D(d_{\text{min}})$. The value of the street parameters are the same as in section 5.6 but we point out that the analytic model is 1-dimensional and thus the street does not have any width (while it is 2 m in the Legion model). In our model, $d_{\text{min}}$ is the minimum required contact duration which may represent the contact setup time plus the download time of a single content item. Contacts with duration shorter than $d_{\text{min}}$ are useless for distributing content. Our results show a good match between the simulations and analytic model. We also see from the tail distribution that after about 5 s there is a sharp drop in probability. This is a strong indication that minimizing the contact setup time $d_{\text{min}}$ is a key performance issue. In fact, this is one of the few system parameters that can be engineered.

In Figure 5.13b we plot the content dispersion $p(x)$ for arrival rates $\lambda = 0.02 \text{ s}^{-1}$ and $\lambda = 0.05 \text{ s}^{-1}$ and for $d_{\text{min}} = 10 \text{ s}$ and $d_{\text{min}} = 15 \text{ s}$. Speed distribution is the same as before and we assume that 5% of arriving nodes are infected ($p_0 = q_1 = 5\%$). Due to symmetry, content dispersion in the reverse direction is simply $q(x) = p(l-x)$ and it is therefore not shown. From the figure we clearly see that content spreads more efficiently at high arrival rates. Also, it is very important to minimize node discovery and contact setup time since the value of $d_{\text{min}}$ affects the dissemination significantly. When the setup time is long (high $d_{\text{min}}$) many of the short contact opportunities are lost. We see that our analytic model is in good agreement with
5.7. A street model

Figure 5.14: An analytic model of the Östermalm scenario in figure 5.1.

the Legion simulation results.

5.7.2 Content dissemination in an urban area

The street model of the previous sections can be extended to an arbitrary topology by interconnecting individual street segments to form a network of streets. This way we can model content dissemination in a scenario such as a larger urban area. The input content dispersion $p_i(0)$ to a street segment $i$ thus becomes the weighted sum of the exit content dispersion $p_j(l_j)$ from all adjacent streets $j$ ($l_j$ denotes length of street $j$).

We have modelled the Östermalm area in central Stockholm, described in section 5.3.1, and studied how content disseminates in the area. The area is approximately $350 \times 380 m^2$ and consists of 29 street segments whose lengths vary between 20 m and 200 m as shown in Figure 5.14. We assume that the arrival process at each entry follows a Poisson process and that all arrival rates are equal, $\lambda_i = \lambda, i = 1, ..., 12$. A fraction $p_{in}$ of the nodes arriving from a single entry are infected with a small content item that is of interest to all susceptible nodes. The arrival rate of infected nodes is thus $\lambda_{ip_{in}}$ and, as before, a susceptible node becomes infected with the content if it has a contact of duration longer than $d_{min}$ with an infected node. We are interested in the content dispersion which equals the fraction of infected nodes in the area in steady state.

Figure 5.15 illustrates the effect of $d_{min}$ on content dispersion for various arrival rates and $p_{in} = 5\%$. It confirms that content spreads relatively well, even at low arrival rates, except when $d_{min}$ is long. When comparing the analytic results with the Legion simulations we notice that the performance of the analytical model is worse. As previously mentioned, the analytic model does not capture node-to-node
crowd interactions but assumes a free flow of nodes. When node density increases nodes are forced to slow down which in turn increases contact duration and therefore also improves the dispersion. Another important issue is that in the analytic model, contacts break at street intersections which is not the case in the simulations. With respect to content dispersion the analytic model can therefore be expected to give pessimistic results.

We have also found that the value of $p_{in}$ does not have a significant effect on the dispersion except at very low arrival rates. Additionally we have showed that replacing the central area of the topology with a square where nodes pause, does not have a significant impact either (pause time is dimensioned such that node sojourn time in the area is the same as before). The interested reader is referred to [47] for details.

### 5.7.3 Transient behaviour

The performance measures extracted from the analytic model are the steady-state averages of the process under inspection. In this subsection we use our simulation model to study what happens when the steady-state is perturbed and when only a single infected node arrives when mobility has reached steady state. We are interested in whether the mobility-assisted content dissemination can sustain content availability when infrastructure is absent and no new nodes are bringing content into the area.

We first run the simulator for a warm-up period until steady state of the mobility has been reached and the total arrival and departure rates for the area have converged. At $t = 0$ (after steady state has been reached) a single stationary node
(an access point) at the entry 1 intersection becomes infected with a content item, all other nodes that arrive into the scenario are susceptible. Nodes will start obtaining the content item, either directly from the access point or from peers that already have it.

For each arrival rate we have conducted 100 simulation runs and in each run we collect the time-series (in 1 s intervals) of the fraction of nodes carrying the content item. In Figure 5.16a we have plotted the average time-series for each of the arrival rates. The results confirm that the spreading of the content item is strongly dependent on the density of the nodes in the area. For $\lambda = 0.05 \text{ s}^{-1}$ approximately 70% of the nodes in the area are carrying the content in steady state and it takes approximately 800 s to reach the steady state average. For $\lambda = 0.01 \text{ s}^{-1}$ the average fraction of nodes carrying the content item is much lower, or just below 20%, and it takes at least 2000 s to reach this steady state average. It is interesting to study what happens with the content distribution process if we turn off the single stationary node that contains the initial content item. In Figure 5.16b we have plotted the same scenario as before except now we turn off the access point when the content dissemination has reached that steady state. For all the arrival rates, steady state has been reached at $t = 2000$ s. It is interesting to note that when the arrival rate is low the content item vanishes from the area after the access point is turned off. At low arrival rates the density of nodes in the area is not high enough to facilitate the mobility-assisted spreading of the content item which in turn vanishes. This can be clearly seen when $\lambda = 0.01 \text{ s}^{-1}$. At higher arrival rates we see however that the spreading of the content is not dependent on the access
5. On the effect of mobility

Figure 5.17: Fraction of runs where content is resident plotted as a function of time. A single infected node arrives at \( t = 0 \).

point and it becomes residential in the area as long as there are new nodes to which it can be passed before disappearing from the area. In other words the spreading process results in a virtual storage effect as the content item resides in the area although there is no infrastructure support and nodes are coming and going.

To further strengthen our assertion of a virtual storage effect we have studied what happens when there is only a single node bringing the content into the area of interest. In this simulation scenario the single infected node enters the area at \( t = 0 \) (after the system has reached its initial steady state distribution) and then we study the fate of this content item. In particular we are interested in seeing if it vanishes and if so, when that happens.

In Figure 5.17 we have plotted the fraction of runs where the content is still resident in the area as a function of time. For \( \lambda = 0.05 \text{ s}^{-1} \) we see that in roughly 20% of the runs the content disappears within 500 s. It is interesting however for the \( \lambda = 0.05 \text{ s}^{-1} \) case that all the runs that have the content after 1000 s will also have it when the simulation ends. This further strengthens our posit that there is a virtual storage effect in the area and that if the content manages to spread initially to a critical number of other nodes, then it will be resident in the area due to this effect. We see the same behaviour for \( \lambda = 0.04 \text{ s}^{-1} \) and \( \lambda = 0.03 \text{ s}^{-1} \) although fewer runs achieve this effect in these cases, or 68% when \( \lambda = 0.04 \text{ s}^{-1} \) and 54% for \( \lambda = 0.03 \text{ s}^{-1} \). For \( \lambda = 0.02 \text{ s}^{-1} \) we do not see this effect clearly which indicates that the arrival rate and the node density is not sufficiently high to maintain the content resident; hence it slowly "leaks out". From the \( \lambda = 0.01 \text{ s}^{-1} \) case we can clearly deduce that this virtual storage effect is not present.
5.8 Conclusion and discussion

In this chapter we have studied how pedestrian mobility affects the connectivity of wireless systems. In particular, we focus on identifying effects due to details in the operational-level mobility process, such as distribution of target speed, node arrival process, node-to-node interactions and specifics of the scenario in which mobility occurs.

Our approach is simulation based and we use the Legion Studio pedestrian mobility simulator. Legion Studio is an agent-based simulator that uses state-of-the-art pedestrian mobility models that realistically capture node interactions and the structure of the space in which mobility occurs. To the best of our knowledge, Legion Studio is the most advanced and realistic simulation model available for micro-level pedestrian mobility and it has, as far as we know, not been used by others for evaluating mobile communication systems.

We have statistically analysed the empirical distributions for the connectivity metrics and how sensitive they are to changes in mobility input parameters. Our findings show that changes in micro-level mobility input metrics do not affect the connectivity graph significantly. The empirical distributions of the connectivity metrics are relatively insensitive to changes in input parameters, showing only a modest change in mean values and retaining their basic shape. This is positive since if the statistics were highly sensitive to modest changes in input parameters, capturing them by a model would be hard and it would be difficult to identify valid parameters for the model. Comparing connectivity metrics across scenarios however shows more difference and this is particularly reflected in the link duration and path duration metrics. This suggests that when modelling mobility, accurately capturing the scenario and its structure is more important than a detailed estimation of input mobility parameters.

We have analysed the empirical distributions of the connectivity metrics and found that there is low correlation between the inter-contact time and link duration. This suggests that the connectivity process of a node can be modelled as a renewal process where the inter-contact time and link duration are drawn independently from the corresponding probability distributions that fit the scenario. An important direction for future work is to identify whether the probability distributions that fit the scenarios we consider also apply to other scenarios. Furthermore, it is important to study how parameters of the probability distributions for the connectivity metrics can be estimated from the input mobility parameters (such as speed, node density, scenario size). We also show that node-to-node interactions in dense scenarios only have a small effect on connectivity and they lead to an increase in link durations which is likely to improve performance for most applications.

Our street mobility model enables us to analytically study the performance and feasibility of mobility-assisted content dissemination in an urban area. Our results show that opportunistic content distribution performs well, even at relatively low arrival rates. We compare the analytic results with Legion simulations to asses the effect of the free flow assumption in our mobility model. This comparison confirms
5. On the effect of mobility

our previous observation; node interactions lead to longer contact durations that tends to give a better performance for opportunistic networking approaches. Our analytic model therefore underestimates the dissemination performance.

Finally we show that if the node density is high enough, the mobility can sustain a virtual storage effect where the content becomes resident in the area even though only a single node is carrying the content initially. A critical parameter for opportunistic content dissemination is the minimum duration that is required for a useful contact. We have confirmed that if node discovery and connection setup takes a long time, system performance is significantly affected to the worse. Many of the contacts that arise among pedestrians are short and therefore it is important to utilize these short contact opportunities for dissemination. This further suggests that if content is large, breaking it into smaller chunks may increase delivery probability since then it can be downloaded in a disjoint manner over different contacts.
Part II

Design, implementation and evaluation
Chapter 6

System design

The performance study in the previous chapter suggests that opportunistic peer-to-peer content dissemination can perform well in urban environments if nodes cooperate. There are however many issues that need to be addressed and solved in an actual system design which is the topic of the current chapter. This chapter introduces PodNet: a system architecture for opportunistic content distribution. We provide an overview of the system and discuss the two different application domains: the fixed Internet and the wireless ad-hoc domain with opportunistic node contacts. The main focus of this thesis is on content distribution among nodes in the wireless ad-hoc domain. It is however important to realize that the PodNet design allows for seamless distribution of content between the wired Internet and the wireless ad-hoc domain. Hence this chapter gives an overview of the full system design.

6.1 System overview

The general PodNet system supports content distribution in, and between, two domains, as shown in Figure 6.1. Content can be generated by servers or hosts in the Internet domain as well as by mobile devices in the ad-hoc domain. The Internet and ad-hoc domains are linked by gateways that assist in disseminating content between domains and perform any necessary translations or proxy services.

PodNet imposes a hierarchical structure on contents. Nodes publish entries on feed channels. A feed channel logically groups together related contents and it consists of a number of entries that contain, among other, the actual data object of interest which we refer to as an enclosure. Nodes express interest in feeds by subscribing to them and the PodNet system then implements the delivery of published entries to feed subscribers, both in the Internet and ad-hoc domains. This structure is based on the publish/subscribe paradigm [27] and our system can therefore be seen as an opportunistic publish/subscribe system.

In the Internet domain, entries can be disseminated from publishers to sub-
6. System design

Figure 6.1: Overview of the PodNet system.

Subscribers using IP multicast and in the ad-hoc domain, content spreads by sharing, which is implemented by a solicitation protocol in which a node solicits entries for one or more feeds from a peer (a peer node can either be a mobile device or a proxy gateway that implements bridging between the ad-hoc domain and the Internet). Feeds and entries contain a number of meta-information fields such as a globally unique ID, author, date and time of last update. The meta-information is primarily used to facilitate searching, filtering and unique matching of contents. The content structure in the system thus allows for ease of searching and a higher hit rate of content queries than if they were made for individual unstructured contents.

The publish/subscribe paradigm is well suited for content-centric networking and it has characteristics that are highly attractive for opportunistic networks with intermittent connectivity. In particular, it decouples publishers and subscribers such that a subscriber does neither need to know who is the publisher of the content nor connect to it in the network. Successful delivery of content is not dependent on the original publisher being up and running; as long as the content is available in the network, either at other subscribers or at caching nodes, a subscriber has a probability of obtaining it. This decoupling also facilitates an asynchronous communication model with loose delay constraints that aids in coping with the dynamic network topology and opportunistic node contacts in the wireless domain. Finally, it does not rely on particular nodes which facilitates a decentralized implementation that is mandatory in the wireless ad-hoc domain and highly desirable in the Internet domain for performance, scalability and fault-tolerance.
6.2 Content structure

Content addressing and organization adopts and extends the content structure of the Atom Syndication Format [74]. This format has primarily been used for publishing web-feeds and podcasts on the web. This content structure is however quite generic and allows for more use cases than what has commonly been tried and it also maps nicely to the publish/subscribe semantics of our system. Contents are grouped into feeds. A feed is an unlimited container for entries that contain the actual data objects of interest. Each feed can have multiple entries published at different times by different entities. Both feeds and entries have associated meta-data. Each feed must contain a permanent globally unique ID assigned by the creator, a title and a timestamp that indicates the latest update. A feed can also contain other optional meta-information such as author, subtitle and category. Similarly, each entry must also contain a globally unique ID, a title and a release timestamp. The feed and entry identifiers are URI's (Uniform Resource Identifiers) which facilitates flexible naming and allows a variety of existing naming mechanisms to be used or new ones to be applied. An entry can optionally have a range of other elements including zero or more enclosures. An enclosure is a single file attachment and would typically be an audio, video, or text file. To be able to efficiently transfer enclosures over the opportunistic contacts, we divide the enclosures into chunks, small data units of fixed size, which can be exchanged with high probability during a single contact of limited duration. Chunks allows the downloading of a previously incompletely downloaded entry to be resumed from the same node or any other node that also has the entry or parts of it. They are indexed starting from 1 and we extend the Atom format to include chunk information for enclosures. If a chunk is only partially received from a peer (e.g. due to lost connection) it is discarded.

6.3 Wireless ad-hoc domain

Nodes in the wireless ad-hoc domain are generally mobile devices, equipped with a radio that can operate in ad-hoc mode and establish contacts with other mobile nodes in their direct communication range. We assume that nodes are altruistic and share content entries they have downloaded with their peers. When two nodes are within communication range they associate and exchange content according to a solicitation protocol. In the ad-hoc domain, content therefore spreads opportunistically from node to node in an epidemic manner with susceptibility given by the popularity of each feed.

Node middleware

In our design, the mechanisms for the opportunistic content distribution system are implemented in a middleware in the mobile nodes. The middleware allows different types of applications to be implemented on top of the basic opportunistic content
distribution service and its purpose is therefore to abstract away the complexities of the underlying system from the applications. In particular, the middleware is responsible for discovering neighbours, coping with none or limited end-to-end connectivity, sporadic contacts and limited contact durations. Moreover, it implements the matching, downloading and storing of content entries. In other words, the middleware implements a session layer that defines the structure of content and abstracts away the network details from applications.

Figure 6.2 illustrates our middleware design and the main system components. Applications access the services of the middleware through the API that it exports. The API implicitly defines the content structure for applications and it allows them to publish/subscribe to content feeds. The design is composed of a set of modules that implement the API, content solicitation on behalf of the applications, service discovery and the solicitation protocol. The architecture also contains a convergence sub-layer for cross-layer interaction, particularly with the underlying radio link such as 802.11/Bluetooth. Their architectures are quite different and thus the session layer architecture abstracts most of the details of the underlying radio and the heterogeneity of the networks away from the applications. The middleware assumes an underlying transport layer that preserves message boundaries, provides flow control and process-to-process communication above an optional network layer. The system design does thus not assume a traditional network layer with point-to-point unicast routing. Contents disseminate in the network by means of node mobility, sharing of local contents and a receiver-driven solicitation protocol. Messages are
delivered on a best-effort basis with no guarantee that entries on a particular feed will be delivered orderly to all receivers.

6.3.1 Interface

The API module implements the programming interface that applications use to access the services of the middleware. The API of our system is inspired by the Java Message Service (JMS) publish/subscribe API [46]. JMS was however designed for wired networks where dedicated brokers implement message delivery. The discovery of feeds also relies on centralized directory service. In the ad-hoc domain, central servers for performing these functions are not available. Instead, both resource discovery and message distribution are performed distributively with servers being replaced by nodes. Thus in addition to standard publish/subscribe/notify functions, the API needs to provide functions that allow applications to discover and create new feeds.

6.3.2 Synchronization & discovery

The synchronization manager module processes content from applications and solicits contents on behalf of them. If the local content database contains data that matches a subscription, the content is delivered immediately to the application. The manager prioritizes content solicitations such that different applications get a fair share of the network resource.

The discovery module is responsible for both neighbour and service discovery; i.e. it discovers wireless neighbours that are running the service and decides which of these are feasible to associate with. Each node advertises its existence to wireless neighbours with beacon notifications that contain the following information:

- **Node identifier** - A URI that uniquely identifies the node.

- **Content revision number** - A simple revision counter for the local content database. The revision number of a node is incremented whenever new content is added to the database. This helps peers to determine if resynchronization might be beneficial in case that nodes remain in range for longer durations or if they meet again after some time away. In particular, two nodes only need to re-associate if at least one of them has obtained new content since they last associated.

- **Feeds Bloom filter** - The list of local feeds with available contents in the form of a Bloom filter. A remote node can compare the feed id’s of its subscriptions to the Bloom filter to deduce if the local node has any content of interest and therefore deduce whether it wants to associate or not. Bloom filters are space-efficient data structures that provide a set-like representation of elements, requiring only a fraction of the space needed for a corresponding set with the actual elements.
Bloom filters trade space for accuracy since false positives can occur with some probability; a membership test returns a value of true but the element is not a member of the corresponding set. False negatives are however not possible. After receiving the Bloom filter, the node tests the ID’s of its subscribed feeds against the filter. Occasionally, a false positive will result in a request for non-existing content. This is however not a serious issue compared with the benefits of using bloom filters: they allow us to include a large amount of information about available contents in a possibly single, small node advertisement message. Often these messages are broadcast and the size of this message is therefore limited by the size of a single link-layer broadcast frame.

How beacon notifications are implemented and transmitted to other nodes may depend on the underlying radio. Therefore the discovery module is split across the main session layer and the convergence sub-layer. Typically, one would send out periodic beacons including the notification but some radios include advanced support for neighbour and service discovery, such as the Service Discovery Protocol (SDP) in Bluetooth.

### 6.3.3 Transport module

The transport module performs session management and implements a request-reply protocol to download and discover available contents at a peer. Protocol messages are in XML format with the `message` element being the kernel of a protocol message. A protocol message has a single `node-id` element containing the ID of the message source and each message has a unique element that determines its type, given by one of the following message types: `request`, `reply` and `reject`. All other elements of a protocol message are child entries for the header fields associated with the message type.

**Session management**

When a feasible peer has been discovered by the discovery module, the transport module is notified which sends a `request` message to the peer to initiate a unilateral session for downloading. The request contains either a query for a particular feed entry, or for meta-data to discover content availability. The peer sends a `reply` message, establishing the session and replying to the query. Each download session thus consists of a client node sending request messages and a server node sending reply messages (or `reject` if the server is unavailable). The server is stateless with each reply message being independent of any previous requests. Processing a request only consists of verifying that the requested contents or meta-data exist and then to deliver them.

Content solicitation in our system is entirely pull-based. At the client, a typical session alternates between `discovery` and `download` states. In the former state, the client node queries the server for content-meta information whereas it downloads contents that match the subscriptions of applications during the latter state. With
this approach, each node has full control of the contents it downloads and decisions are based only on the client state with the server being stateless.

In general, a node can have multiple active sessions simultaneously with the node being either a client (when it is downloading) or server (when it is uploading) in each session. Note that the system does not explicitly enforce any mechanism to share download time between sessions; we simply rely on the mechanisms of the MAC layer to share the radio channel fairly. Ungraceful session termination (e.g. when nodes move out of range) is handled by a soft-state timer; if there is no activity from the peer for a certain time, the session is closed and any allocated resources are freed up.

Content solicitation

A request message is used both for downloading and discovery of contents. Discovering which previously known feeds or entries are available at a peer node is done efficiently by the use of bloom filters. As described in section 6.3.2, nodes maintain a Bloom filter with the id’s of available feeds. In addition they also maintain a Bloom filter for each feed that includes the id’s of available entries. When a node receives a request with an empty XML Bloom element, it delivers a corresponding bloom filter in a reply message. After receiving the filter, the client node tests the ID’s of its desired feeds or entries (or partially downloaded entries) against the filter. Then it sends individual requests for each entry that it wishes to retrieve. Listing 6.1 illustrates an example request message that requests four chunks from an entry that has an audio/mpeg file as an enclosure.

A Bloom filter does not allow for iterating through the elements it contains and thus it cannot be used to discover previously unknown contents at a peer. The protocol therefore implements additional mechanisms for discovering previously unknown feeds and new entries on already known feeds. A request message can either contain an empty feedlist element or an empty entrylist element to indicate that it wants to receive the list of available feed or entry id’s at the peer. The selector element of a request message can also be used to solicit meta-information for contents that match a particular selection criteria given by a content...
selector. A content selector is a string whose syntax is based on a subset of the SQL conditional expression syntax [46]. A node that receives a request message with a selector as top-level element of a request, evaluates the selector on the attributes of each of its available feeds. The feed elements for which the selector evaluates to true are delivered in a reply message. Similarly, a selector specified inside a feed element will be evaluated against all entries of the specified feed and only those entry items that evaluate to true are delivered. An empty selector will match all feed/entry elements and those attributes not specified in the selector evaluate to true by default. Since nodes can have large content libraries, specifying a selector when discovering feeds can significantly reduce the amount of meta-data delivered in a reply message.

6.3.4 Discussion

The architecture described here assumes that each node shares content available to the applications on its device and it specifies a protocol and mechanisms for efficient downloading of this content between nodes in the ad-hoc domain. The PodNet system does not mandate or specify a particular content caching or forwarding mechanism in addition to the interest driven solicitations given by the application subscriptions. The content structure and solicitation protocol specified in our design is however general such that applications can implement their own caching or replication strategies.

6.4 Internet domain

Nodes in this domain have fixed Internet connectivity, generally over a wired connection or a wireless last hop, such as when associated with an 802.11 access point or a 3G connection. In the Internet domain, content is disseminated from publishers to subscribers using multicast which is both scalable and effective. PodNet uses the SLIM single source multicast protocol [37] for best-effort delivery of multicast datagrams. Before describing the design of the publish/subscribe delivery subsystem of PodNet we therefore give a short overview of SLIM and its basic functionality.

6.4.1 SLIM overview

Several factors have prevented IP multicast [23] from being a widely available network service in the Internet. In particular, the PIM-SM [30] IETF proposal requires complex mechanisms to be standardized and universally deployed such as for routing, address allocation and source (or rendezvous point) discovery. Therefore, single source multicast has been proposed as a simpler alternative to the any-source model of [23]. In single source multicast the delivery is reduced to a one-to-many paradigm only and instead of identifying a multicast group by a single class-D IP address, a multicast channel is identified by a pair $<S,C>$ of IP addresses, $S$ is the IP address of the source, which is the root in the multicast tree, and $C$ is the source specific
channel identifier chosen by the source such that the pair \(<S,C>\) is globally unique. In practice, \(C\) is often a class-D IP address (and a requirement in some protocols). Through a simpler service model, single-source multicast avoids most of the complexity required for realizing any-source multicast. At the same time, single-source multicast can be used as a building block for any-source multicast applications [54].

SLIM is a single-source multicast protocol that extends previous proposals (in particular [38]) by including support for incremental deployment and, apart from a control-plane daemon process for topology management, does not require any multicast specific functionality in routers. The design of SLIM benefits from increasing availability of three datapath facilities in routers: a multicast capable forwarding engine, a general-purpose classifier and dynamic tunnelling. SLIM self-configures itself over the Internet, and includes mechanisms for traversal through firewalls and NATs while exploiting available layer-2 multicast services without intervention of network administrators. It also specially addresses the issue of delivering multicast data to end-systems that reside multiple hops away from a multicast capable router. SLIM therefore provides single source multicast as an elementary network service upon which more elaborate applications (possibly with multiple data sources) can be built.

Hosts interested in receiving multicast data obtain a session description containing \(<S,C>\) and send a join control message towards the source \(S\). The join message is sent with the router alert IP option [55] and a special IP protocol num-
6. System design

ber (we use 242 in our prototype implementation) which causes it to be intercepted and processed by SLIM capable routers. Routers that do not recognize the protocol number do not process the packet any further and should forward it as usual. If the router receiving the \texttt{join} message is already on the distribution tree, the message is suppressed and the router inserts a new forwarding entry in its classifier that maps packets with IP source and destination addresses \texttt{<S,C>} to the new tree branch. Otherwise the \texttt{join} message is forwarded upstream in the same manner until it reaches the source or a router that is already in the dissemination tree. The \texttt{join} message contains, among other, the IP address of the last SLIM capable router that processed the message along with the TTL value of the IP header at that router. By comparing the TTL value in the \texttt{join} message with that of the IP header a SLIM capable router can discover non-capable routers on a tree branch and dynamically create an IP-in-IP tunnel between SLIM the capable routers. Figure 6.3 illustrates an example of a host joining a multicast channel. Forwarding state needs to be periodically refreshed by \texttt{join} messages and it will be removed if three consecutive messages are not received. The \texttt{join} messages therefore create and maintain a logical overlay tree of SLIM capable routers over the physical network topology.

6.4.2 PodNet content dissemination

Each feed that is available in the Internet domain has a \textit{designated host} that is responsible for maintaining and making a \textit{feed description} available. A feed description is an XML document that contains session- and meta-information for the feed and its entries.

When a host on the Internet wants to subscribe to a PodNet feed, it first obtains the feed description from the designated host (via http or any other means). The feed description contains (among other) a multicast channel for the feed. As a result of a subscribe operation, the PodNet middleware on the host issues a join message for the multicast channel which results in the host being grafted onto the multicast tree for the feed. Thus feed subscribers in the Internet domain are the leaves in the multicast tree for the feed.

When a new entry is published on the feed, the publishing host transmits the entry via standard TCP unicast to the multicast source for the feed. The source notifies the designated host which updates the feed description. Then the source delivers the entry on the multicast tree. This is shown in Figure 6.4 where host H1 sends an entry to the source S which delivers the entry on the multicast tree, completing the publish operation. PodNet thus uses the source as a multicast relay to implement a multi-source multicast service using single-source multicast as building block.

In PodNet, feed entries should in general be available for access even after their original publication. Therefore the designated host should archive, or provide an indirection to, entries for later access. This provides mobile nodes that are disconnected at the time of publication, a means to access content when they connect to Internet.
6.5 Gateways

In addition to providing for a feed description, multicast and archiving services, the designated host can also implement services such as authentication, access control and pre-processing of content. These are however application functionalities that are not a part of the PodNet distribution platform but could be implemented on top of it if required. Therefore they are not discussed further in the context of this thesis.

6.5 Gateways

Gateways provide the interconnection between the Internet and ad-hoc domains and their purpose is to make content available between domains. When compared to the mobile nodes, gateways generally have ample storage, power and network capacity. Gateways can therefore use their resources for pre-fetching and caching content to assist in disseminating content to the resource limited mobile nodes.

Generally, a gateway is a node that has both a fixed Internet connection and a wireless interface that can operate in ad-hoc mode, such as 802.11 or Bluetooth. It implements both the Internet and ad-hoc parts of the PodNet system and provides a transparent proxy service to the nodes in the ad-hoc domain. In particular, it implements the PodNet content solicitation protocol used in the ad-hoc domains (described in section 6.3) and the mobile nodes therefore communicate with a gateway in the same manner as with other peers. This facilitates simplicity in the design and efficiency since the wireless nodes do not have to switch between modes (e.g.
from ad-hoc to infrastructure mode) to obtain content from the Internet.

Gateways can subscribe to popular feeds and cache or pre-fetch content (and meta-data) that is likely to be requested by nodes in the ad-hoc domain. When a mobile node queries a gateway for content that is not available in the gateway’s cache, the gateway will contact the designated host for the feed to fetch content on behalf of the mobile node. This is performed transparently to the mobile node.

When a mobile node contacts a gateway the meta-information for the subscribed feeds of the mobile node is updated to match the feed description from the designated feed host. In other words, the mobile node discovers those entries that have been published in the Internet domain that it is missing. Since content can also be generated in the ad-hoc domain, the mobile node may be carrying entries that have not been published in the Internet domain. The gateway can discover and obtain these entries by querying the mobile and publish them in the wired domain on its behalf. We however point out that feeds can be created in the ad-hoc domain and it is not necessary for the creator to specify a designated host for the feed on the Internet.
Chapter 7

Implementation

We have implemented a prototype of our middleware design for the ad-hoc domain described in the previous chapter. This chapter describes implementation details of the mobile middleware and applications that use the services provided by the middleware.

7.1 Overview

Our implementation runs on the Google Android Platform and we have successfully deployed it on HTC Hero smartphones. The system is implemented in Java and currently our code consists of 55 classes and roughly 3500 lines of code. The implementation is currently based on 802.11 in ad-hoc mode. The current Android Java libraries (version 2.3) do however not support the ad-hoc mode of 802.11 although it is supported by both the driver and the hardware interface on the HTC Hero device. Therefore, our implementation requires the device to be run in privileged user mode (i.e. rooted mode) so that the interface can be reconfigured to run in ad-hoc mode.

7.2 Software modules

Our implementation consists of software modules that implement the functionality of the corresponding modules in our design in Figure 6.2. We will now describe some implementation details of these modules.

Interface

The middleware is implemented as an Android service which runs in the background and uploads and downloads data from peers that it discovers. Client applications

---

1We intend to support Bluetooth in the near future
can bind to the service and communicate with it by means of remote procedure calls (RPCs) through the pub/sub interface that it exposes. A client application wishing to receive a notification when an entry matching one of its subscriptions is downloaded, needs to implement and register a callback function that the service uses for notification. The interfaces for the service API and the application callback function are shown in listing 7.1. The remote methods exported by the service through the IServiceAPI interface are executed synchronously, thus blocking the local thread at the caller. In the service process, a method call is executed in a dedicated thread chosen from a pool of threads that is maintained by the Android system. The callback method in the IClientCallback interface is however executed asynchronously (specified by the oneway keyword) and therefore the service does not block when it notifies a client application.

**Discovery module**

The discovery module is implemented as two threads. One thread periodically broadcasts hello messages on a well-known UDP port and a listener thread waits for incoming hello messages from other nodes. The discovery module maintains a contact history cache along with the revision number for each peer in the cache. When a new peer is discovered, the discovery module notifies the transport module which initiates a download session with the peer. If a peer, already in the contact history cache, is seen, the transport module is notified if the peer has obtained new contents since the last association or if there are new subscriptions locally.

**Transport module**

The transport module implements both the client and server sides of a download session. The solicitation protocol is currently implemented on top of a simple transport protocol that implements message boundaries on top of TCP. The server side implementation listens on a socket and spawns a new session thread for each
client. Similarly, if multiple nodes are in communication range the transport module can create a separate client thread for each session. Currently we set the maximum number of concurrent client and server sessions to 6 in total (3 for each). If a new node tries to associate when the maximum number of sessions is reached, the server sends a reject message.

Content database

The content database of the system is implemented as an Android Content Provider. Meta-information for all available feeds and entries is stored in a SQLite database and this information is accessible to all applications on the device through the ContentProvider and ContentResolver Android Java classes. The enclosures themselves (i.e. data files) are however not stored in this content database but in the corresponding Android Content Providers. Images, audio and video contents are for example stored in the Android MediaStore content provider. Thus, all media content published or downloaded by our system is available to all applications in a standard Android manner.

7.3 Applications

We have currently implemented two different applications on top of our middleware system, an opportunistic media blog application and a personal profile exchange application.
Oppportunistic media blog - With this application, a user can take photos or videos with her phone, attach a caption or a short message, and publish the content as a new entry on one or more feeds. The media file for the photo or video becomes the enclosure of the new entry and the caption is stored as a corresponding xml element within the entry element.

Personal profile sharing - With this application, users at a conference or a meeting can share personal profiles (i.e. electronic business cards) and vote on events such as the conference venue. The application subscribes to a conference and a voting feed and the profile of the user is published on the former and a voting on the conference venue is published on the latter. The application thus implements the user interface needed to configure the user profile and view the profile of others. It also implements an interface for voting and a simple mechanism for counting votes that the device has received and a GUI for these. Figure 7.1 illustrates the application GUI.

These applications serve the purpose of evaluating both the design and implementation of our middleware and they have also been demonstrated publicly as a proof-of-concept [107, 58]. In particular, these applications illustrate how our system can be used for different types of content centric applications. In the following chapter we present, among other, results from an experimental evaluation of our system using the media blog application.
Chapter 8

Evaluation

In this chapter we evaluate our system and dimension system parameters. The evaluation uses three methods: analysis of connectivity traces, simulation and experimentation.

8.1 Chunk size dimensioning

Currently, there are two technologies for wireless opportunistic networking that have widespread deployment in mobile devices, 802.11 and Bluetooth. Of the two, 802.11 has both higher bandwidth, longer communication range and shorter contact setup times and will therefore give better performance in most scenarios. The downside is that 802.11 generally requires more energy and some mobile devices are only equipped with a Bluetooth radio. Therefore it is important that our system is dimensioned such that it also gives acceptable performance on a Bluetooth implementation and therefore we choose Bluetooth connectivity traces to dimension the chunk size parameter.

The purpose of dividing an entry into fixed-size chunks is to allow disjoint downloading of content from different nodes and to make the most out of all contacts, even those that are short because of node mobility. With large chunks there is an increased risk of reduced performance when partially downloaded chunks are dropped. This issue is important for short contacts or devices with low bit-rates. In particular, if chunks are too big, starvation can become a problem when nodes are not able to download a single chunk during a typical contact. A small chunk reduces the amount of wasted download data due to partially delivered chunks but it also increases overhead since chunk lists in message headers may become large and there is a small header associated with each chunk. Therefore we seek to have a chunk size that is as large as possible under the constraint that both peers in a contact of a typical duration obtain at least one chunk each with high probability.

In dimensioning the chunk size we have studied two measurement traces from the Crawdad database [21]. We refer to the traces as Rollernet [95] and Toronto [90].
8. Evaluation

Both traces log the contacts and contact durations of mobile Bluetooth devices carried by humans. We selected these particular traces because they are comprised of data from users with high mobility, resulting in a challenging environment for opportunistic networking. These traces also scan more frequently for neighbours than many other Bluetooth traces and therefore have higher contact resolution. Now we briefly describe the context and methodology of capturing the traces. For a further description we refer to [95] and [90] respectively.

**Rollernet**: Small sensors, equipped with a Bluetooth radio (iMotes), were distributed to 62 participants of a large (about 2500 participants) rollerblade event in Paris. Periodically, each device scans the devices in its vicinity and logs encounters with iMotes or other external Bluetooth enabled devices (mostly mobile phones). The duration of the experiment was about three hours and the dataset consists of 38,405 unique contacts of non-zero duration with a mean duration of 27 sec.

**Toronto**: This dataset consists of contact logs captured by pedestrians in the city of Toronto, Canada. The traces were collected inside two shopping malls and while riding in the Toronto subway system. Two Palm Tungsten T PDAs, with PalmOS version 5.0, captured contacts with external Bluetooth devices. The total number of contacts in the dataset is 526 and the mean contact duration is 27 sec.

The Toronto experiment measured the average throughput between two Bluetooth devices to be 185 kbps. If we assume that the radio channel capacity is shared equally between the peers and ignore protocol headers and propagation delay\(^1\), the

---

\(^1\)Protocol header sizes are about two orders of magnitude smaller than the chunk sizes we consider and propagation delay is negligible compared to the transmission time of a chunk.
transfer time of two chunks is \( t_{tr} = \frac{2 \text{chunksize}}{\text{throughput}} \). Figure 8.1a shows the empirical tail distribution of the contact duration and Figure 8.1b shows the probability that a contact is longer than \( t_{tr} \) for different chunksizes for both traces. Although the transfer time is a rough estimate it (along with the empirical contact duration distribution) gives us an idea of how the chunk size can affect the system performance. We see that for a chunk size of 16 KB it is likely that two peers will be able to obtain at least one chunk each during a contact (with probability higher than 0.92 for both traces). The recommended chunk size for content in the system is therefore 16 KB. We expect this value to give decent download performance and avoid chunk starvation even when deployed in a setting that has limited throughput and relatively long setup times such as in a mobile Bluetooth environment.

### 8.2 Experimental evaluation

The experimental evaluation is performed on our Android implementation. We measure some key implementation metrics, such as energy consumption and application layer throughput (i.e. goodput), in small and simple static scenario. The experiments are performed on identical HTC Hero A6262 mobile devices. These devices have a 528 MHz Qualcomm MSM7200A processor, a ROM of 512 MB and RAM of 288 MB and a Lithium-ion battery with capacity 1350 mAh. During our experiments, communicating nodes were stationary in an indoor office environment and placed within one meter from each other.

#### Energy consumption

We have measured the effect of our system on the battery life of the device. The Android system sends out an event notification (Intent) whenever the remaining life of the battery changes (in units of 1%). We have created a simple application that registers for these events, and logs the time whenever the battery status changes. This way we can track how fast the battery is drained when various system services and applications are turned on or off. In Figure 8.3a, we compare the battery profile for five scenarios: a) with the 802.11 interface turned off and our system not running, b) with the 802.11 interface turned on in ad-hoc mode but our system not running, and with our system running with the interval between hello messages set to c) 0.1 sec, d) 1 sec and e) 10 sec. All measurements were performed on the same device with no other active devices in range at the same time. During all measurements the display backlight was turned on. This drains the battery faster than in normal mode but prevents the device from going into idle battery saving mode which reduces the comparability of our measurements.

From Figure 8.2 we clearly see that the 802.11 interface significantly increases energy consumption. Running our system (in idle mode, only sending hello beacons) in addition to the 802.11 interface does not add considerably to the energy consumption beyond what is required by 802.11. When beaconing every 0.1 sec-
8. Evaluation

onds\textsuperscript{2}, the battery lasts approximately 40 minutes shorter than when the hello messages are sent every 10 sec.

The high energy consumption of 802.11 is an issue that raises concerns and may hinder deployment of our system. We are currently working on a Bluetooth implementation which may alleviate this issue. Bluetooth was however not originally intended for ad-hoc networking, but rather as a cable replacement technology, and therefore suffers from other issues such as long discovery time and mandatory user intervention when pairing with a device, both of which make Bluetooth ill-suited for mobile scenarios. It is important to point out that this energy issue is not induced by our system per-se but an issue with the 802.11 radio and thus something that is of general concern for opportunistic networking based on 802.11. This is one of the main motivations for why we have started working on a energy-efficient radio architecture for opportunistic networking which will be discussed in chapter 9.

### Solicitation protocol profiling

We have profiled our implementation of the solicitation protocol to verify correct behaviour and to assess performance. For our measurements we have instrumented the code with hooks where we stamp the system clock (which provides millisecond precision). During a measurement run we turn off logging and collect the measured timestamps into a list which is printed to a file after the code section being measured.

\textsuperscript{2}This is the beacon period commonly used by 802.11 access points.
8.2. Experimental evaluation

Figure 8.3: (a) Profiling results for the mean feed and entry discovery delays. Each group of two bars contain results with one feed in the content database (left) and 100 feeds in the content database (right). (b) The mean goodput of a download session when the number of concurrent clients is varied between one and three.

has completed running. This minimizes the effect of any I/O operations due to logging or measurements on our results.

As described in section 6.3.3, a typical download session consists of three steps: 1) the client discovers available feeds at a server by sending a request for a feedlist (or the corresponding Bloom filter), 2) then it discovers available entries for a given feed in a similar manner and 3) finally downloads the entry of interest. In Figure 8.3a we show the mean feed discovery and entry discovery delay (steps 1 and 2). We have conducted measurements for three different enclosure sizes and for each enclosure size we conduct one set where the content database only contains the actual feed and entry of interest (left-side bars) versus the case when the database has 100 other feeds available (right-side bars). For each measurement we conduct 10 runs and in the figure we show the mean value and the standard deviation. The results confirm that the total discovery delay (i.e. the sum of the of the feed discovery and entry discovery delays) does not depend on the size of the downloaded enclosure. When the number of feeds in the content database increases, the feed discovery delay increases due to an increase in the number of bytes transmitted in the feedlist in the reply message and processing delay at the server. We see also that the entry discovery delay remains the same since the number of entries on the feed of interest is the same in all experiments.

Our implementation supports multiple concurrent download sessions and in Figure 8.3b we show the average goodput (i.e. application level throughput) of a session when the number of devices concurrently downloading is between 1 and 3. Our measurement setup is as follows. Between one and three nodes (referred to as clients)
are within range of a single node (referred to as server) which publishes a single entry on a feed that the client nodes are subscribing to. When the client nodes receive the first hello message sent by the server after the entry publication, the clients see that the server has new content and therefore simultaneously associate with it. The client nodes discover the entry and then download it and we measure the goodput \( G \) of each session as \( G = \frac{B}{T} \) where \( B \) is the total number of bytes transmitted and \( T \) is the duration of the download session, i.e. the elapsed time from when the client discovers the node until it receives the full entry and enclosure. The figure shows that although the implementation supports concurrent download sessions, the throughput is still reduced because access to the wireless interface is serialized by the operating system at the server. The performance drop is however clearly less than 50% which is the minimum performance drop when each node only serves a single client.

8.3 Simulation evaluation

In addition to the Android implementation of our system, we have also implemented the core modules of our design for the OMNeT++ simulator, using the framework in chapter 4. A mobile node in the simulator implements the service discovery, synchronization and transport modules of our architecture described in chapter 6.

Simulation scenario

The application we consider is mobile peer-to-peer file sharing among people in an urban area. The files being shared are assumed to be audio or small video files and this is reflected in our settings for feed popularities and entry sizes. Nodes in our simulations have a simple transport layer and a CSMA/CA MAC layer with a maximum frame size of 1500 bytes and a bit-rate of 185 kbps (matching the measured value in section 8.1). Node transmission range is 10 meters and each node only downloads from one neighbour at a time (while multiple uploads are possible).

Mobility model

The mobility scenario consists of \( 3 \times 3 \) city-blocks in central Chicago (see Figure 8.4a) for which we generate mobility traces with the UDel Models [56] mobility simulator. The UDel mobility simulator is based on empirical models extracted from surveys on work-time statistics, urban planning and traffic engineering. Sidewalks, streets and buildings restrict the movements of pedestrians and their mobility depends on their activity, e.g. whether a node is at work, going to/from work or having lunch. This mobility model therefore mainly captures human mobility at the strategic and tactical levels. In comparison to the Legion studio mobility model used earlier in the thesis, the UDel model results in a coarser representation of mobility. The benefit is however that we run simulations for a longer time, for a
8.3. Simulation evaluation

Figure 8.4: (a) The simulation topology of $3 \times 3$ city-blocks in central Chicago. (b) Empirical artist popularity distribution of the last.fm website and a fitted Zipf law.

larger area and for more nodes. We simulate 30000 nodes for one day but we only export traces for the fraction of nodes participating in the opportunistic content distribution; a maximum of 700 nodes between 07:00 and 13:00 in our setting.

Content generation

To simulate the system performance for a file-sharing application we need to make assumptions on which feeds a particular node is interested in and how many entries it initially has for each of its subscribed feeds. To generate these in a realistic manner we assume that the main type of content being exchanged in the file sharing application is music, since for this type of content we can quantify the feed popularity distribution. We assume that a particular feed represents an artist and we assign feed subscriptions to nodes according to artist popularities. To obtain an empirical distribution for artist popularities we have crawled weekly charts, published at the last.fm website [59], and collected artist popularity data for a period of one year. Last.fm is a large community website (over 30 million users) which builds a profile of the musical taste of users. Last.fm weekly publishes a chart with the top 420 most popular artists, measured by the number of users that listened to each artist during the week. In Figure 8.4b we have plotted the artist popularity as a function of the artist rank-order. We fitted a Zipf distribution with parameter $\alpha = 0.368$ to the collected data which is seen to give a good match for roughly the 380 most popular artists. The sharp drop at rank 380 is due to the limited sample size of the data; the popularity of artists only appearing in a few of the weekly charts is underestimated since we only have data for the most 420 popular artists each week.
In our simulations there are 100 available feeds, initially with 15 entries each. Each node selects $F$ feeds it subscribes from the Zipf popularity distribution. For each selected feed, the node selects $E$ of the 15 initial entries as part of its initial content library. During the simulation, each node publishes a new entry randomly on one of its subscribed feeds, every $P$ seconds. This could for example be a new file obtained off-line (via infrastructure) by the user. In table 8.1 we list some of our simulation parameters and their values.

**Dissemination performance**

The performance metric we study is the the mean number of completed entries at the end of the simulation. On a per node level, it represents the expected library increase of a node and thus provides a measure of the utility gained for the system users.

In Figure 8.5 we have plotted both the private and total number of completed entries per node as a function of $N$; the number of nodes participating in the content distribution. The left figure shows results for a mean entry size of 3 kB while in the right figure it is 3 MB. The 3 kB file size could for example represent an event based notification system while 3 MB is representative for sharing of audio files. The results shown are the per-node averages, weighted by the lifetime of each node in every simulation run (we are simulating an open system where nodes can enter and leave during the course of the simulation). The solid black line represents the average number of downloaded content entries per node and it confirms that the system performance scales nicely with the number of nodes; performance improves when more nodes participate.
8.3. Simulation evaluation

Figure 8.5: Number of entries completed per node for a mean file size of 3 kB (a) and 3 MB (b).

Effect of caching

A previous study of opportunistic podcasting [63] suggested that soliciting and caching public content, i.e. content that nodes are not privately interested in, results in improved system performance since content availability in the system is higher. Other works have also studied different content caching and replication strategies for opportunistic content dissemination as we reviewed in section 2.3 in chapter 2.

We have implemented three different public solicitation strategies in our simulation model to compare with system performance without caching where the content spreading is only driven by the node interests. In our simulations with caching, each node selects three feeds that it is not privately subscribed to and solicits content for these feeds. Initially the public cache of all nodes is empty and public content solicitations always get lower priority than private content solicitations. We consider three different public solicitation strategies based on content popularity.

- **Most popular** - Public feeds are randomly selected from the Zipf popularity distribution.
- **Random** - Public feeds are selected uniformly at random from all available feeds.
- **Least popular** - Public feeds are randomly selected according to their inverse popularity in the popularity distribution, i.e. the least popular feed gets the popularity of the most popular feed etc.
- **No caching** - No public content is cached or solicited and system is based on direct interest dissemination only.
8. Evaluation

The motivation behind the random and least popular strategies is that they might improve system fairness and give less-popular content a better chance to spread since private feed subscriptions already implement a most popular strategy.

The lower family of lines in Figure 8.5 shows the average number of downloaded private content items while the upper family of curves shows the total number of content items downloaded. The total number therefore also includes the items that the user downloads only for caching. The striking result from the figure is that caching and soliciting public content does not lead to any significant performance improvements. In the case of the smaller file size (Figure 8.5a), improvements are marginal. For the larger file size (Figure 8.5b), the public caching actually results in a slightly diminished performance compared to a system with no public cache. This is in spite of the fact that public content soliciting increases the total number of completed entries by roughly a factor of two in our setting. Also, this behaviour does not seem to depend strongly on which public solicitation strategy is used.

Figure 8.6 shows how the public caching affects the total number of downloaded entries of each individual feed for the case $N = 700$. The top figures show results for the most popular feeds and the least popular feeds for a mean entry size of 3 MB while the bottom figures show the same for a mean entry size of 3 kB. With a mean entry size of 3 MB, public caching diminishes the dispersion of almost all feeds. Only a few of the least popular feeds experience a marginally better dispersion with the least popular caching strategy than with no caching. Even for feed 1, the most popular caching strategy performs worse than with no caching. When nodes are busy downloading public entries, other contact opportunities that could have led to the sharing of private entries are blocked and lost. This is due to the relatively low bitrate and large entry size. This blocking does not seem to have a strong effect for a mean entry size of 3 kB and in general, uniform and most popular caching help spreading feeds since more private entries are completed when compared to no caching. Although least popular caching increases performance for the least popular feeds it performs poorer than no-caching for the three most popular feeds. The performance increase for the least popular feeds that is seen by the least popular strategy is therefore achieved at the expense of other more popular feeds. We have also conducted a set of simulations where we study the effect of having initially a non-empty public cache and they show no significant differences from the presented results.

Our results on the effects of public caching are somewhat in contrast to those presented in [63] and we believe that there are various reasons to this. We use a more realistic mobility model, as well as a more realistic model of the mac- and physical layers. The simulation model presented here also implements an actual solicitation protocol and content popularities are based on measured data and it evaluates the system on a larger scale, both in terms of the number of nodes and content feeds. The performance metric studied in [63] was based on the content delivery delay, a metric that does not capture any trade-off between performance and the overhead introduced by caching in terms of energy and storage usage of nodes. In our study on the entry delivery delay we have found that for the 3 MB mean entry size, delay
is increased by the use of public caching. For the smaller entry size (3 kB) public caching only slightly reduces delay for a few of the most/least popular feeds when using most/least popular soliciting. Most feeds however experience no significant reduction in delivery delay or even an increase.

The main conclusion from our study is that public soliciting and caching increases resource consumption for downloading and storing significantly but does not lead to any significant increase in system performance. Due to this we do not include any particular caching strategy in our architecture; the system is based on direct interest sharing only which has the potential to give good spreading performance when a reasonable number of nodes participate in the system. This design decision leads to a simple architecture and avoids some of the privacy, integrity and incentive issues that arise when nodes are carrying content on behalf of others. We however point out that our architecture does not prevent applications to implement their own caching strategies.

Various other strategies than the ones we evaluate here have been proposed, as listed in chapter 2. Some of these caching strategies are based on exploiting...
8. Evaluation

the users social networks to try to identify nodes they frequently meet and solicit contents for these [20, 101, 10]. The elapsed time between meetings of nodes in the same community is however often long, on the order of tens of minutes or hours. We believe that a common case for users in urban areas is that they are regularly connected to infrastructure networks where they have access to vast amount of contents. One of the main benefits of a opportunistic content distribution system is then that it allows users to access some contents while on the move between these occasions of Internet connectivity, such as when travelling to/from work. We therefore believe that the goal of caching should be to bring what is immediately available in a close neighbourhood to the interested user and in this respect the solutions proposed in [40] and [82] seem particularly interesting. As a conclusion we note that our study of content caching is not conclusive and remains an important direction for future work. A comparative study of caching proposals that takes into consideration overhead and resource consumption is needed.
Chapter 9

Energy-efficient radio architecture

In the previous chapter we saw that the energy consumption of the wireless interface on a smartphone can draw significant amount of energy, draining the battery fast. This is a practical concern that can restrain the deployment of mobile opportunistic systems. In this chapter we evaluate the potential energy savings that can be achieved by using a dual-radio subsystem. With such a system a low power radio is used to discover neighbours and contents, and the high power radio is only woken up for downloading if the peer has contents of interest.

9.1 The need for an energy efficient radio subsystem

Since opportunistic content distribution is mainly targeted at mobile users with battery powered devices, it is of great importance that the protocols and mechanisms used are energy efficient. Although recent smartphones are powerful devices with advanced networking and multimedia capabilities, battery capacity is still a scarce resource.

Mobile handhelds today have commonly two available radios that can operate in ad-hoc mode: Bluetooth and 802.11. Bluetooth requires less power at the expense of a shorter communication range (approx. 10 m) and lower bitrate (around 2.1 Mbit/s for Bluetooth 2.0). The neighbour discovery process of Bluetooth constitutes a major drawback for being used in opportunistic networking. Bluetooth uses a frequency hopping spread spectrum scheme and therefore, scanning for neighbours is an operation that consumes significant power, prevents normal data flow and can take as long as 10 seconds. Thus neighbour discovery is typically performed infrequently (less than once every minute). As a result Bluetooth performs badly in mobile scenarios, missing out a large fraction of potential contacts [106]. In comparison, 802.11 provides a higher bitrate (a raw rate of 54 Mbit/s in 802.11g), a longer communication range (approx. 100 m) and neighbours can quickly be discovered by using broadcast beacons. The downside is that it requires more energy. Interestingly however, 802.11 has a better energy per bit ratio than Bluetooth (from a pure
9. Energy-efficient radio architecture

Energy (bit viewpoint), making 802.11 more suitable for transmitting bulk data. In summary, 802.11 is a better candidate for opportunistic networking but minimizing its energy consumption is one of the key challenges for enabling opportunistic communication as a viable communication mode.

It is known that the wireless LAN interface is often responsible for a large fraction of the energy consumption [89] of mobile devices. In the previous chapter we have confirmed that this also applies to the HTC Hero platform on which we have deployed our system. Figure 8.2 shows clearly how the 802.11 radio interface (in ad-hoc mode) affects the battery life on the HTC Hero smartphone. When the interface is turned on, the battery life is reduced to only 25% of what it is with the interface turned off. This is despite the fact that no application data has been transmitted or received via the interface. Once the interface is turned on, it consumes relatively high power regardless of being in a transmit or receive state since waiting to catch a signal to decode in idle state consumes almost as much power. Thus, broadcasting UDP beacons periodically at different rates does not significantly affect the energy consumption of the device. This suggests that reducing or eliminating the idle energy cost of the 802.11 interface may be a promising strategy to reduce the overall energy consumption and prolong battery life. As a minimum requirement, the battery of a smartphone running an opportunistic networking service should last at least a full day so that it can be recharged during night. Minimizing the energy consumption of the networking subsystem is therefore one of the key challenges for enabling opportunistic communication as a viable communication mode.

In this chapter we study and evaluate the potential performance gains that can be achieved by equipping mobile handhelds with a dual radio system. A high-power, high-bitrate radio is used for data transfer and a low-power, low-bitrate radio is used as a control channel for performing neighbour discovery [86]. The high power data radio can thus be suspended and only woken up when a neighbour is discovered by the low-power control-channel radio. This way, the high-power radio is only used for transmission/reception of data and its idle-mode cost is significantly reduced or, at best, eliminated. Our work is based on a simulation study and, after discussing related work in the next section, we present our simulation framework for a dual radio subsystem in section 9.3. Section 9.4 presents our initial results on the potential energy savings that can be achieved when using a dual radio architecture in an opportunistic content distribution system.

9.2 Related work

In [28], Feeney and Nilsson measure the energy consumption of an 802.11 interface in ad-hoc mode and provide a detailed energy profile. They show that the interface consumes significant energy in idle mode and that the ad-hoc mode is not as energy efficient as the infrastructure mode. This is mainly because the infrastructure mode has more efficient energy saving mode in which it can rely on the access point to synchronise and buffer data for nodes that are sleeping to save energy. In [29], Feeney
and Wilkomm describe their implementation of a framework that can be used to simulate the energy consumption of a mobile node in the OMNeT++ simulator. This framework contains a detailed energy model for the IEEE 802.11 wireless LAN standard as well as for other technologies. Our dual-radio simulation framework uses the aforementioned energy framework to model node energy consumption.

The *Wake on Wireless* system [86] was the first to propose a dual-radio system in the context of mobile handheld devices. The prototype implementation used a simple low-power radio as a control channel for discovery and to wake up the 802.11 radio when a neighbouring node has data to send. They show that their system can significantly increase the battery lifetime of such devices. In [80], Pering et al. further show, by prototype measurements, that significant energy can be saved by using a low-power Chipcon CC100 radio or Bluetooth for discovering 802.11 access points. These small-scale (2-3 nodes) evaluations focused on infrastructure systems where the low-power radio is used to wake up the high-power radio when it is in reach of an access point, or when another node wants to communicate with the local node via the access point. In contrast, we focus on evaluating the efficiency of a dual radio system in an opportunistic setting where no fixed infrastructure can be assumed and nodes are not synchronized. Our study is based on simulations where we employ and adapt detailed simulation models for both energy consumption of wireless radios and node mobility.

CoolSpots [79] proposed a dual-radio Bluetooth/802.11 architecture to switch between radio interfaces with respect to traffic intensity. The goal was to use Bluetooth at low application workloads and switch to 802.11 when load increases and bandwidth availability in Bluetooth saturates. Both the Wake on Wireless and CoolSpots design focuses on, and assume the availability of, infrastructure networks with access points. These systems are therefore not suitable in a distributed ad-hoc scenario.

In [50], the authors consider a dual-radio system for delay-tolerant networking. They rely on GPS for clock synchronization and examine how to dimension the duty cycles of the radios for a given traffic load and contact statistics with minimal energy consumption as an objective. In contrast our work does not consider DTN routing but focuses on how a dual-radio system can reduce energy consumption in the context of opportunistic content distribution. We assume neither GPS availability, which itself consumes a non-negligible amount of energy [15], nor reliable mobility prediction, but we rather focus on how incorporating application semantics in the discovery on the low-power control channel can reduce energy consumption. Currently we assume that the discovery radio is always on, but the availability of synchronized clocks or asynchronous wake up mechanisms [105] for radio-specific power management may provide additional energy savings (that is however currently not the focus of our work).
9.3 Simulation framework

In this section we describe our design and implementation for supporting multiple controllable radios per node in OMNeT++. Our implementation is based on, and extends, the MiXiM framework and it is non-intrusive and does not break pre-existing code. The description in this section assumes a dual-radio approach but the design is general and not just limited to two radios per node. Our code is available as a MiXiM branch at http://github.com/olafur/mixim.

9.3.1 Design Overview

Figure 9.1 shows the main modules comprising a mobile node equipped with two different radio subsystems: a primary low-power, low-bitrate network interface card (NIC) (primaryNic) and a secondary high-power, high-bitrate NIC (secondaryNic). In our design, the two radios co-exist at the node without the knowledge of each other. When running a simulation one would typically have a global connectionManager module for each radio type that manages the connections for each interface type (i.e. two connectionManagers if all nodes have identical dual radios).

Our design does not require modifications to any protocol layers outside of the NIC which allows for flexibility in choosing where to implement the logic for controlling the NICs. On the one hand, this logic can be implemented in the application (as in Figure 9.1). On the other hand it could be placed in a lower layer, and therefore the application would not necessarily have to know that multiple radios are used.

Just as in MiXiM, a NIC is a compound module that consists of a physical

Figure 9.1: An example dual radio node.
9.3. Simulation framework

layer submodule and a submodule that implements the multiple access protocol (MAC). One of the main features of our design is that we have extended a NIC to be controllable in the sense that it can be suspended or woken up by sending control instructions to it.

Each NIC can be in one of the following three states: TURNED_ON, SLEEPING or TURNED_OFF. A TURNED_ON NIC has full functionality, as currently implemented in MiXiM. A NIC in TURNED_OFF or SLEEPING mode is not active for transmission or reception of packets, but can be turned on (or woken up) when requested by upper layers. The main difference between the TURNED_OFF and SLEEPING states is that they can be configured with different energy consumptions and that there can be different latency for turning on versus waking up.

Apart from the radio and networking related sub-modules, each node also has a blackboard, a battery and a mobility module. The blackboard serves as a notification mechanism for modules to signal internal state changes to other modules. It is for example used by MiXiM modules such as the mobility module to notify changes in position and the battery module to signal host failure due to battery depletion. With our extensions, a NIC can be controlled by external modules via the blackboard and a NIC also publishes any changes in its on/sleep/off state to the blackboard. We will now describe in more detail our implementation and extensions to individual components.

9.3.2 Controllable NIC

In our design, we extend a NIC to include a NicController submodule in addition to the standard mac- and physical-layer modules. Moreover, all the submodules of a NIC are extended to implement the IControllable abstract interface shown in Listing 9.1. The NicController has the following main responsibilities: 1) Receive control commands to the NIC from external modules (via the blackboard). 2) Ensure that the NIC submodules are suspended and awakened in correct order. 3) Simulate the delay when waking up or turning on a NIC and 4) publish state changes of the NIC on the blackboard. An external module thus controls a NIC by publishing one of the Controls defined in IControllable on a special control category on the Blackboard. Since the NicController is subscribed to this category it receives the control messages from the publishing modules. When the NicController receives a control message, it forwards it to the mac and physical layers of the NIC. Since all the submodules of a controllable NIC implement the IControllable interface, they invoke the appropriate protected member function that handles the control.

When both the MAC and physical-layer modules of a NIC have changed their states, the NicController publishes the new NIC IControllable::Status on the blackboard. The blackboard thus allows us to inform all interested modules simultaneously about a state change, and then each module can decide whether and how to treat this information.
9. Energy-efficient radio architecture

class IControllable {
public:
    enum Controls { TURN_ON, SLEEP, WAKE_UP, TURN_OFF };
    enum Status { TURNED_ON, SLEEPING, TURNED_OFF };
    virtual bool isOn();
    virtual bool isSleeping();
    virtual bool isOff();
protected:
    virtual bool turnOn() = 0;
    virtual bool sleep() = 0;
    virtual bool wakeUp() = 0;
    virtual bool turnOff() = 0;
};

Listing 9.1: Abstract interface for a controllable module.

9.3.3 Extending MAC and PHY

When a new node is created in MiXiM, the initialize methods of all the node
submodules are invoked. In MiXiM, a NIC does not have any on/sleep/off states
and it is initialized in a state that corresponds to our TURNED_ON state. In our
framework a NIC can be in any state when a node is created (it is configurable)
and therefore we need to override the default MiXiM initialization of mac and phy
modules.

In order not to break any existing code we create new mac and phy modules
that extend the corresponding MiXiM modules. In these new modules we defer
the standard initialization by overriding the initialize function of the parent
class. In the new initialize function we now only read configuration parameters
but defer the radio-state initialization which is instead executed when a Control
message is received.

Listing 9.2 shows how we extend the physical layer of the MiXiM Energy Frame-
work (PhyLayerBattery) with a new implementation. The new module only needs
to override the initialize/finish functions and the control handler of the orig-
inal physical layer (handleUpperControlMessage), all other functions can be left
as-is. In addition, the new physical layer implements the IControllable inter-
face (as described before) and the state change routines are now invoked from the
overridden control handler.

When a NicController receives a TURN_ON message it is passed to the MAC
layer after some simulated delay. For the CSMA based MAC implementations in
MiXiM (i.e. Mac80211, CSMA and CSMA802154) the MAC cannot start sensing
until the physical layer has been turned on. Therefore the control message is passed
down directly to the phy which in turn invokes turnOn. This method registers the
NIC with the global connectionManager module for that particular radio type
and initializes the radio which starts drawing a current from the battery. When
the phy has turned on, it sends a TURNED_ON message up to the mac which can now
start sensing the channel. Then the MAC sends the TURNED_ON message up to the
class PhyLayerControl  
  : public PhyLayerBattery, public IControllable  
{
  public:  
    virtual void initialize(int stage);  
    virtual void finish();  
    virtual void receiveBBItem(int category, const BBItem *details, int scopeModuleId);  
  protected:  
    virtual void handleUpperCtrlMessage(cMessage* msg);  
    virtual bool turnOn();  
    virtual bool turnOff();  
    virtual bool sleep();  
    virtual bool wakeUp();  
};

Listing 9.2: Definition of a controllable physical layer module.

NicController which publishes the message on the blackboard.

A TURN_OFF control message will result in the inverse behaviour. First the mac empties all send and receive buffers. Then the phy unregisters from the connectionManager, stops drawing current and finally a state update is published by the NicController when the TURNED_OFF message has been passed from the phy to the mac to the NicController.

The handling of WAKE_UP and SLEEP messages is similar. The main difference is that the delay in waking up is different from that of turning on and that the current drawn in SLEEP state is different from that in TURNED_OFF state.

9.4 Performance evaluation

Evaluation scenario

We consider the energy savings achievable with a dual-radio subsystem in the context of our opportunistic content distribution system. We have extended the OMNeT++ simulation implementation of our system (that was used in section 8.3) such that it can take advantage of a dual-radio subsystem when available. In our system model we use the low-power radio for neighbour and content discovery and only turn on the high power radio when downloading a content entry. The high power radio is based on the 802.11 MiXiM model and the low-power radio uses the 802.15.4 MAC implementation in MiXiM. For the 802.11 radio, we use the energy specification from the MiXiM Energy Framework and for the low power radio we use the energy specifications from the Texas Instruments ZigBee-ready RF transceiver (CC 2420).

When using a dual-radio approach, the beacon messages and the messages for discovering available feeds and entries are exchanged on the low power radio. Since the beacons do not contain much information, they fit into a single 802.15.4 broad-
cast frame. Only the downloading of the actual content items is performed over the high-power interface, which is suspended otherwise. Thus when the local node discovers a remote node that has one or more content items of interest, the local node turns on its high power interface. To ensure that the remote node also turns on its high power interface for uploading, we use a simple control protocol over the low power interface. The local node sends a turn-on control message to the remote node. When the remote node receives the message it turns on its high power interface (if not already turned on). When the interface at the remote node has been completely turned on, it sends a turned-on message on the low-power interface to the local node. This message also contains the MAC address of the high power interface at the remote node and now the local node can request the discovered content items on the high power radio. When the local node is finished using the high-power radio it sends a turn-off message to the remote node to gracefully signal completion. Ungraceful session closure, such as when nodes move out of range or if the initiating node crashes, is handled by a soft state timer. This simple control protocol could of course be enhanced with the goal of achieving further energy savings. The nodes could for example synchronize their clocks over the low power radio and duty cycle the high power radios. There are however various issues to consider, in particular since nodes can have active sessions with multiple nodes simultaneously. The current goal of our work is however not to propose an advanced power control scheme but mainly to assess, by a first approximation, the potential energy savings of a dual radio architecture.

For our evaluation we have set the communication ranges of both radios to equal values which is highly desirable in a dual radio system. If the low power discovery radio has shorter range, then valuable contacts might be lost. If the low power radio however has greater range then it becomes difficult to use the low power radio for neighbour discovery since a neighbour on the low power radio is not a neighbour on the high power radio. We are aware that 802.11 radios usually have a longer communication range than 802.15.4 radios. With a simple external power amplifier the range of an 802.15.4 radio can however easily be increased at the expense of a slightly increased energy consumption and a lower bitrate [32]. For our evaluation we have set the communication range to 10 m for both the radio types. The mobility model we use in our evaluations are the Legion Studio Subway and Östermalm models described in section 5.3.1. In both mobility scenarios we assume that there are 10 available feeds in the area, each feed containing 10 entries. Every device is subscribed to one feed upon entering the observed area, and its content database is initially populated with five random entries (out of ten entries available) on the feed. Thus, throughout its lifetime in the simulation, each node strives to obtain the rest of the entries that belong to its subscribed feed. Entries have a mean size of 10 KB, and a standard deviation of 2 KB.
9.4. Performance evaluation

Results

We investigate the performance of the opportunistic content distribution system both in terms of energy consumption and goodput (i.e. application throughput). We compare the system performance when using a dual radio subsystem and when using a single high-power 802.11 radio only. All metrics are normalized with respect to performance for the single radio case and plotted showing 95% confidence intervals. The mobility model represents an open system and therefore it is important that the metrics are further normalized with respect to the node sojourn time in the simulation. The goodput of a node is simply the number of bytes downloaded divided by the lifetime of the node in the simulation. We only count bytes of fully downloaded content items so the goodput is a measure of the system usefulness for the users, i.e. how much content it can provide. The energy consumption of nodes is also normalized with respect to the node sojourn time.

As we can see from Figure 9.2 both the Östermalm and the Subway scenario, although different in their characteristics, exhibit similar behaviour when a dual radio approach is used. We see a significant decrease in the energy consumption, which is due to energy savings in the process of node and content discovery. For both cases the dual radio requires only 20% of the energy used with a single high-power radio. At the same time, we observe that this decrease in energy consumption comes at a price of a reduction in the amount of information obtained by the devices. In a dynamic environment where opportunities arise on the go and last only while two nodes are in communication range, the delay associated with turning on a suspended high-power interface (set to 2 s in our simulations) comes at a cost of lost contacts. It can happen that by the time the high-power interface of a node is
brought up from off state, the peer carrying the content of interest has moved out of range and the opportunity for obtaining the content is lost. This is reflected in a performance drop in goodput. The goodput is reduced to approximately 50% of a single radio goodput for both mobility scenarios.

In our simulations, waking up from a sleep state does not take any time but instead the the high-power interface consumes a small amount of energy (10% of the energy consumption when the interface is turned on). The comparison between a sleeping mode and the off mode shows that the sleeping mode exhibits higher goodput values (more than 20%) compared to complete high-power suspension while the difference in the amount of energy consumed in the two modes is negligible. Interestingly, we even find that the energy consumption when using a sleep state is slightly lower than with an off state for the Östermalm (figure 9.2a). Due to the wake-up delay of a completely suspended high-power radio, the probability of lost contacts increases. Thus, it may happen that the high-power radio is brought up after the peer of interest has already moved out of range. This also means that the high-power radio will not be turned off by the remote peer and therefore the node needs to wait certain amount of time before a soft-state timer expires and the radio is suspended (its value is set to 4 s in our simulations).

9.5 Conclusion and discussion

In this chapter we have studied the potential energy savings that can be achieved when nodes participating in opportunistic content distribution are equipped with a dual radio subsystem. We presented our simulation framework and our initial performance results which indicate that significant amount of energy can be saved with a dual radio system. These energy savings however come at the price of some reduction in application throughput.

This chapter describes work in progress and we are currently looking further into various issues such as the effect of communication range and mobility. Both of the mobility traces we use are characterized by high node density and high mobility which leads to many contact opportunities for nodes. These traces therefore represent a challenging scenario for a dual radio architecture and we believe that more energy savings can be achieved in a scenario that also captures the more stationary facets of human mobility (such as when riding on a bus or staying at work). This is one of the issues we are currently looking into.
Chapter 10

Conclusions

In this thesis we have studied opportunistic content distribution among mobile wireless nodes. Content is disseminated from node to node by exploiting the contact opportunities that arise when two mobile nodes come into direct communication range. This communication mode has several benefits.

- Opportunistic content distribution enables dissemination of content between mobile nodes without relying on infrastructure. This is beneficial where infrastructure is absent, overloaded, unreliable, expensive to use, censored or limited to certain users or contents.

- Today, content availability on a mobile device is strongly coupled with availability of Internet access. When hooked up to the Internet we have virtually unlimited access to information while if we are disconnected we have very limited access, if any, to information that is non-local to the device in hand. Opportunistic content distribution introduces a new way of accessing contents by bridging this dichotomy in connectivity.

- Given the current exponential growth of mobile data traffic it is likely that the capacity of infrastructure systems may become a major bottleneck in the near future. The scaling properties of opportunistic communication is different than the scaling of infrastructure systems and performance improves with the number of nodes. Opportunistic communication can therefore be used in parallel to offload the infrastructure. With respect to content availability, scaling also comes naturally as popular content is likely to be available at many nodes in the system.

Using both analytic models and simulations, we have studied the feasibility of opportunistic content distribution and explored how performance is affected by cooperation when nodes share contents and user mobility.

By extending and applying stochastic models from mathematical epidemiology we confirm that node cooperation is essential. However, limiting the number of
times a node shares a content item only slightly reduces performance when compared to unlimited sharing. This observation might be used by nodes to reduce the number of transmissions and conserve energy.

Our evaluation of the effect of mobility is mainly based on a simulation study where we use Legion Studio, an advanced pedestrian mobility simulator, to model node movements. We study how connectivity metrics, such as the duration of a contact and the rate of contacts, are affected by mobility parameters such as the target speed of nodes and arrival process. We find that the connectivity metrics are not very sensitive to modest changes in the mobility parameters but that capturing the overall scenario in which mobility occurs is important since this affects performance. We also show that the free flow assumption, commonly used in mobility models, can result in an underestimation of performance. When nodes interact, they tend to slow down, resulting in longer contact durations and in mobile communication, longer contact durations are usually correlated with better performance. Then we present an analytic model to study the spreading of content in an urban area, modelled as a grid of streets. Our main finding is that content can spread well even when user density is relatively low. However, to obtain good performance it is important to minimize the neighbour discovery and contact setup time to successfully exploit the short contact opportunities that arise when moving in an urban area. Currently there are two wireless technologies that have widespread deployment and can be used for ad-hoc opportunistic networking, 802.11 and Bluetooth. Node discovery and contact setup time in 802.11 can be relatively short but the power consumption is a concern which can make 802.11 less favourable than Bluetooth. Bluetooth on the other hand suffers from long node discovery and our evaluation shows that this can significantly affect dissemination performance. These are important issues to address to fully unleash the potential of opportunistic networking. A comparison between our analytic model and simulations suggests that performance could be even better than predicted by our analytic model due to the aforementioned free flow assumption used in our model.

In summary, the feasibility evaluation shows that popular contents can be well served with interest driven opportunistic epidemic spreading where nodes store and share the contents that they are privately interested in. The system design we present in this thesis is based on this premise. Our middleware architecture enables a publish/subscribe content dissemination among mobile nodes without infrastructure support. We have specified and described core components of the system such as the structure and naming of contents, a solicitation protocol for discovering and downloading contents in a wireless ad-hoc domain and an API. Our system design also supports seamless dissemination of content between the wired Internet and the wireless opportunistic domain. On the Internet, published entries are delivered to subscribers by means of the SLIM single source multicast protocol.

A fundamental design decision of our system is that it is based on a purely best-effort approach and we acknowledge that not all contents can be discovered and delivered by our system. The goal of our system is not to provide Internet-
like availability of contents in an opportunistic network but rather to provide users with access to some contents without requiring Internet connectivity. The evaluation in this thesis suggests that popular contents can be well served in this manner and we believe that there are many common scenarios and applications where opportunistic communication can provide good performance, such as in spontaneous meetings among a group of people, at sporting events or when traveling with public transportation. Furthermore, in a city, certain contents are likely to be available at many nodes since many people are usually interested in local news, traffic information, weather and content from local websites, television or radio programs. Including and embracing content popularity in the design leads to a simple architecture which inherently focuses on performing well for the common case. In contrast, complicated protocols would be needed to locate and deliver less popular contents or contents from a particular node.

We have implemented our opportunistic content distribution system on the Android platform and in a simulator environment. We have performed a small scale experimental evaluation with the Android implementation where we measure the energy footprint of our system and profile the solicitation protocol. We have also dimensioned system parameters using connectivity traces from mobile nodes and evaluated system performance on a large scale with our simulator implementation that utilizes a realistic mobility model of pedestrians in an urban area.

Future work

The work presented in this thesis has many possibilities for future work. Below we discuss some issues that we consider particularly interesting, some of which we have already started pursuing.

Applications and experimental evaluation

The true test of an architecture is whether it can meet the demands of a wide range of applications, both those that we can presently anticipate as well as some that might emerge in the future. We are currently looking into other applications than those already presented in the thesis, such as applications based on opportunistic user voting, location based applications where contents may be associated with a particular position or region and using opportunistic content distribution in an industrial or educational environment. As part of this we are interested in performing an experimental evaluation of our system using a larger number of devices than can be achieved with small scale laboratory experimentation.

Security

In this thesis we have not addressed the important issues of security, integrity and privacy. Some of the main security challenges associated with our system relate to verifying the integrity of content items and preventing system abuse and
spamming, for which many solutions already exist. The main complication is that in an open opportunistic environment where no central authority can be assumed, infrastructure-based and hard cryptographic solutions can be difficult to apply. If one however assumes that users can regularly access a trusted authority, message verification can be deferred and many of the security concerns can be resolved. For some scenarios, particularly urban areas, it can usually be assumed that users will regularly have Internet access and that they can then obtain public cryptographic keys needed for verifying message integrity. For immediate security measures in the opportunistic domain, mechanisms built on reputation [64] and trust systems [96] seem to provide an attractive approach.

**Energy efficient radio**

The work in chapter 9, on the efficiency of a dual radio subsystem for opportunistic networking, is still in its early stage. We intend to complement our evaluation study by looking at more mobility scenarios and study further the effect of the communication ranges of the different radios. We believe that 802.11 is the most promising wireless technology that is currently available for opportunistic networking but its energy consumption is an issue that may hinder the deployment of wireless ad hoc systems.

**Content replication**

In section 8.3 we evaluated our system when, in addition to the interest based sharing, nodes replicate and cache content on behalf of others. The caching strategies we used were based on content popularity and our results show that replicating content on behalf of other nodes with these strategies can significantly increase resources consumption for downloading and storing but does not lead to any considerable increase in system performance. In section 2.3 we reviewed some recent proposals, but to the best of our knowledge these have not been compared thoroughly or implemented in real systems. Devising a content replication mechanism that increases system performance while still being light on resource consumption remains an important direction for future work.

**Seamless connectivity**

The main focus of this thesis is on content distribution in the wireless opportunistic domain. There are however various interesting issues to explore on the boundary between the ad-hoc wireless domain and the Internet domain and how to provide users with seamless connectivity when moving between domains. We are for example interested in studying whether caching in gateways can improve system performance and whether gateways should participate in some form of overlay to infer the mobility patterns of nodes and utilize these for content pre-fetching.
Appendix A

Absorption analysis of continuous time Markov chains

We denote by $Q$ the transition-rate matrix of a continuous time Markov chain (CTMC). For an absorbing CTMC with a single absorbing state the matrix $Q$ can be decomposed as

$$Q = \begin{pmatrix} T & A \\ 0 & 0 \end{pmatrix} \quad (A.1)$$

where $T$ is the square matrix of transient-to-transient state transition-rates and the column vector $A$ are the rates from the transient states to the single absorbing state. If the initial state distribution among the transient states is given by the vector $p_0$, the mean time to absorption for the chain is

$$E[T_{\text{absorption}}] = p_0(-T)^{-1}1 \quad (A.2)$$

where $1$ is a column vector of ones. For a more thorough discussion see for example [11].

Equation (A.2) assumes that there is a single absorbing state. If this is not the case, all the absorbing states can be collapsed into a single state before extracting the matrix $T$ from $Q$. Numerically solving (A.2) therefore consists of the following steps

- If the Markov chain is more than one-dimensional we renumber the states with a 1-dimensional linear index.
- Collapse absorbing states into a single state.
- Renumber states such that $Q$ is on the form (A.1).
- Extract the matrix $T$ from $Q$ and solve (A.2).
Appendix B

Kolmogorov-Smirnov test results

This appendix contains results from the Kolmogorov-smirnov tests in chapter 5. The Kolmogorov-smirnov test is a nonparametric statistical test that is frequently used to compare sample data with a reference probability distribution. In our tests we compare the sample data with six reference probability distributions: exponential, log-normal, gamma, weibull, pareto and rayleigh. A lower KS-statistic indicates a better fit.

<table>
<thead>
<tr>
<th></th>
<th>Exp.</th>
<th>Log-norm.</th>
<th>Gamma</th>
<th>Weibull</th>
<th>Pareto</th>
<th>Rayl.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Österm. LD</td>
<td>0.27</td>
<td>0.20</td>
<td>0.27</td>
<td>0.25</td>
<td>0.30</td>
<td>0.57</td>
</tr>
<tr>
<td>Österm. ICT</td>
<td>0.08</td>
<td>0.06</td>
<td>0.02</td>
<td>0.01</td>
<td>0.04</td>
<td>0.37</td>
</tr>
<tr>
<td>Subway LD</td>
<td>0.15</td>
<td>0.07</td>
<td>0.13</td>
<td>0.11</td>
<td>0.09</td>
<td>0.49</td>
</tr>
<tr>
<td>Subway ICT</td>
<td>0.21</td>
<td>0.07</td>
<td>0.04</td>
<td>0.01</td>
<td>0.08</td>
<td>0.56</td>
</tr>
<tr>
<td>Österm. PD, h=2</td>
<td>0.23</td>
<td>0.15</td>
<td>0.14</td>
<td>0.18</td>
<td>0.24</td>
<td>0.42</td>
</tr>
<tr>
<td>Österm. PD, h=3</td>
<td>0.24</td>
<td>0.18</td>
<td>0.15</td>
<td>0.18</td>
<td>0.24</td>
<td>0.38</td>
</tr>
<tr>
<td>Subway PD, h=2</td>
<td>0.20</td>
<td>0.11</td>
<td>0.16</td>
<td>0.19</td>
<td>0.17</td>
<td>0.66</td>
</tr>
<tr>
<td>Subway PD, h=3</td>
<td>0.27</td>
<td>0.15</td>
<td>0.20</td>
<td>0.23</td>
<td>0.22</td>
<td>0.70</td>
</tr>
</tbody>
</table>

Table B.1: K-S statistics for distributions fitted with simulation results. LD = Link duration, ICT = Inter-contact time, PD = Path duration and h is the path length. All results are from simulations with communication range $R = 10$ m.
## Kolmogorov-Smirnov Test Results

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Österm., h=2</td>
<td>0.10</td>
<td>0.08</td>
<td>0.08</td>
<td>0.09</td>
<td>0.11</td>
<td>0.36</td>
</tr>
<tr>
<td>Österm., h=3</td>
<td>0.12</td>
<td>0.11</td>
<td>0.11</td>
<td>0.13</td>
<td>0.15</td>
<td>0.43</td>
</tr>
<tr>
<td>Subway, h=2</td>
<td>0.10</td>
<td>0.09</td>
<td>0.09</td>
<td>0.10</td>
<td>0.12</td>
<td>0.38</td>
</tr>
<tr>
<td>Subway, h=3</td>
<td>0.12</td>
<td>0.11</td>
<td>0.10</td>
<td>0.12</td>
<td>0.14</td>
<td>0.39</td>
</tr>
</tbody>
</table>

Table B.2: K-S statistics for distributions fitted with simulation results of path duration. All results are from simulations with communication range $R = 30$ m.
Bibliography


[87] Tara Small and Zygmunt J. Haas. The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way). In Proc. of ACM MobiHoc, 2003.


