DEPARTAMENT DE TELECOMUNICACIÓ I D'ENGINYERIA DE SISTEMES

DOCTORAT EN D'INFORMÀTICA INDUSTRIAL I TÈCNIQUES AVANÇADES DE LA PRODUCCIÓ

TREBALL DE RECERCA DE DOCTORAT

Evaluation of Dynamic Routing Protocols on Realistic Wireless Topologies

Estudiant

Roger Baig Viñas

Director

Dr. Daniel Riera i Terrén

Col·laboradors externs

Axel Neumann, Ester López

September 29, 2014

Abstract

Community Networks are consolidating themselves as a valid model to extend the edges of the Internet. As a result of efforts to overcome specific problems in this new model, communities have developed very interesting ideas and solutions in many fields. Nevertheless, the research community has so far paid marginal attention to them. As a consequence, not only is there a lack of references evaluating the performance of the routing protocols for IPv6 in real-life scenarios, but other interesting proposals such as BatMan-eXperimental version 6 (BMX6) remain completely unstudied.

The routing protocol selection is one of the most critical choices any community must make prior to any hardware deployment, in the delicate moment when the community is just starting to form. In our opinion, in such cases, an exhaustive evaluation of the performance of the available routing protocols would ease this selection process.

In an effort to contribute in this direction, this dissertation first analyzes the topology and link characteristics of a well-known Community Network (CN). In a second step, this new knowledge is used to parametrise an emulation environment in order to reflect relevant attributes of a real wireless CN and to study the performance (in terms of protocol overhead and convergence time) of the Babel, BMX6 and OLSR routing protocols for Internet Protocol version 6 (IPv6).

Acknowledgements

Special thanks must be given to Axel Neumman, not only for his invaluable direct contribution to this work but also for being always willing to share his broad knowledge with everyone. Undoubtedly without him the work here presented would never have materialized. It is always a great pleasure to work with you.

Special thanks must be given to Ester López as well. Her contributions has also been essential to make this work possible. Thanks for sharing your research results, ideas and opinions and sorry for being always late.

Many thanks to my supervisor, Prof. Daniel Riera i Terrén, for trusting me. It took us a couple of years, but finally we did it.

Thanks also to Prof. Felix Freitag for his encouragement and Prof. To Leandro Navarro for the guidance and the advice provided and for believing in the Community Networks. To Aaron, Joseph and Mitar for the information about their respective Community Networks and specially for taking care of them. To Judith for her work on some of the figures. To Brian for reviewing the abstract and the introduction.

Last but not least, special thanks to Ramon Roca and Lluís Dalmau for their baby, a marvellous Community Network called guifi.net. Projects like yours show that a better world is possible and give strength to keep fighting for it.

This work is partially supported by the European Community Framework Programme 7 within the Future Internet Research and Experimentation Initiative (FIRE), Community Networks Testbed for the Future Internet (CONFINE), contract FP7-288535.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Community Networks (CNs) are IP-based networks designed, built, operated and maintained by communities of individuals that join together in an effort to, at least partially, satisfy their telecommunication needs and desires. CNs are an emerging model for the Future Internet that enhances the opportunities of local stakeholders to develop community services including local networking, voice, data, Internet access, etc.

CNs are large-scale distributed and decentralised systems with many computing nodes, links, content, services and traffic. They are extremely dynamic and diverse as they are built in a decentralised manner, mixing wireless and wired links with diverse routing schemes and with many services and applications. There are no barriers for the participation as governance, knowledge and ownership is open, with an open peer agreement governing the network. Therefore these networks are not only decentralised but are also self-owned and maintained, and expanded upon by community members, and grow continuously in links, capacity and services provided.

In such contexts of decentralization, continuous modification, and heterogeneity, some of the essential protocols of the Internet become unusable because their assumptions are not fulfilled in this kind of networks. Therefore CNs are very challenging in terms of research and development. Despite the fact that some very interesting and innovative

solutions have been developed and successfully tested in diverse arenas, including socio-economic and technical ones, the academia has regrettably only recently begun to take an interest in CNs, hence, many areas remain unstudied.

The Dynamic Routing Protocols (DRPs) are not an exception. Although many theoretical proposals have been made by the academia to solve the challenges associated with such anarchic networks, where traditional routing protocols are simply not suitable, very few of them have been implemented and, to the best of our knowledge, none of the latter has ever been tested in an environment as stressful as a CN. Simultaneously, many practical proposals have appeared as free software projects in the CNs scenario, some of them using or taking as starting point academic proposals, but others based on totally new concepts and ideas.

Unfortunately, the lack of exhaustive works evaluating and comparing the existing DRPs driving the decision-making process leaves the CNs in a risky situation because it is a crucial decision that conditions the CNs evolution and must be made at the very beginning.

1.2 Aim

In order to help CN activists select a DRP, in this work three of these protocols are analysed under an emulated network environment by replicating relevant topology characteristics of a real CN with tenths of nodes. In an effort to contribute to the normalization of the Internet Protocol version 6 (IPv6) usage, this work is IPv6-only.

1.3 Choices

The following subsections cover and justify the main decisions made in this work.

1.3.1 Metrics

Protocol overhead and convergence time are fundamental characteristics of any DRP and the most common performance indicators use to compare them. Protocol overhead is the quantity of control traffic sent by a routing protocol in order to propagate routing and topology information over the network. Convergence time is the time a protocol takes to become aware of a change in the network and to recalculate and apply all the necessary routes to address this change.

There are many factors that may affect the performance of a protocol. The most important are: the overall network size (total number of nodes), the network diameter (maximum number of hops between most distant nodes), the link quality, links density (number of links per node), number of announced interfaces and of Host/Network Announcements (HNAs) per node. This work focuses on the first three: network size, network diameter, and link quality.

1.3.2 Dynamic Routing Protocols

The work presented focuses on the analysis and evaluation of the implementation of the three DRPs: Babel, Optimized Link State Routing Protocol (OLSR) and BatMan-eXperimental version 6 (BMX6). All three implementations are developed and maintained by community members and made publicly available as free software projects.

Babel is standardised by an Request for Comments (RFC). It is a relatively young and simple protocol that has been completely implemented. It has been exhaustively analysed by the author of the RFC, who is also the main developer of the implementation evaluated in this work. Thus, many references can be found in the literature. It is (up to now) only rarely used in currently existing CNs.

OLSR was started as an implementation of an RFC but soon extra features were added to address shortcomings of the standard that emerged from its deployment and usage in real-life networks. The progressing development of the implementation lead to partial incompatibility with the inspiring RFC. Nowadays OLSR is very popular in CNs and due to its wide usage has become a de facto reference both by the DRP developers and the DRP analysers. Hence it can be frequently found in the literature.

BMX6 is relative new DRP characterised by a radical new approach and incorporates numerous promising features. Due to its novelty, few references can be found in the literature. The developer of this DRP is one of the external supervisors of this work.

1.3.3 Community Network

Despite CNs are technologically agnostic, and despite that optical fibre links are gradually becoming more common, nowadays the vast majority of the links in CNs are still wireless[1]. In the same way, all links in this work are implicitly taken to be Wireless links and CN may be sporadically referred to as Wireless Communities.

In order to feed the emulation framework with realistic scenarios, the relevant network parameters of the Barcelona guifi.net zone has been analysed in detail. Guifi.net is the biggest CN in the world. As part of the tools developed by the community, and one of the keys to its success, the detailed description of the whole network is stored in a database. This information is publicly available via Community Network Mark Up Language (CNML), a specification based on XML. Moreover, most of the guifi.net routers have enabled a *guest account* through which some of the routers' configuration details and statistics are accessible.

We have been deeply involved in guifi.net for more than five years and contributing to the development of the zone under study since the very beginning.

In this work, just the so-called *supernodes* in guifi.net terminology (the core nodes) and their links are considered because these are the only nodes that actively participate in the routing decisions.

1.3.4 Community Network characterisation

Despite the fact that the monitoring information collected and made available in real time at guifi.net's website is very useful for management and planning tasks, it is not

[1] IEEE 802.11 standard, also known as WiFi.

comprehensive and precise enough to fulfil our needs for a characterisation of a link-centric topology. Therefore we had to acquire our own data and process it. My Trace Route (MTR) was the application selected to do the raw data acquisition of the links.

Further post-processing of the raw MTR data was needed and achieved via the development and application of adaptable shell scripts and fed into the emulation environment.

1.3.5 Emulation framework

Common approaches to analyse and evaluate distributed applications are usually given by simulation or by experimentation in laboratory environments. The first option comes with a high level of abstraction and often requires a reimplementation of a given protocol for the chosen simulation environment. This approach has the downside that the simulated protocol significantly differs from an implementation used on real hardware in a CN. The second option of experimentation on real hardware allows the usage of the same implementation as used in CNs but demands either the (costly) acquisition and set-up of many computers and links (which introduces new difficulties like the set-up of a network topology that reflects relevant characteristics of real-life wireless network) or demands the execution of experiments in an existing (and usually productively used) CN.

A third option is given by emulation which is in fact a mix of the first two options where parts of the evaluation environment are simulated and other parts are executed on real or virtualised hardware. This option has the advantage of being cost efficient since multiple node instances can be created as virtual systems executed on a single (but powerful) computer and links between nodes can be emulated with state-of-the art network simulation tools to introduce desired link characteristics. Since the same implementation of the analysed protocol can be used in the emulation, completely identical protocol behaviour can also be expected.

Further advantages of emulation over physical experiments are: experiments are reproducible and require much less effort to set up, scenarios can be pushed to the limits to allow the evaluation of predictable future scenarios (e.g. anticipated network growth), the whole environment can be reset at any time without further consequence, etc. Additionally, in our case, by using the emulation framework, we overcome the problem

of accessing all the real nodes and avoid the risks of performing experiments over a production network.

Emulation techniques may be as extensive as the reconstruction of a whole computer, including the BIOS, with its own kernel and operating system, and even some installed applications - known as virtualisation - or a simple new instance of the hosting kernel, perhaps with an independent file system - known as contextualisation. Since in our case all nodes will run almost the same image (i.e. same kernel and almost the same file system - just a few network parameters will change from one to the next), contextualisation is clearly the best option.

For the objective of this work various open-source network virtualisation and emulation frameworks such as Mininet[1], Mesh Linux Containers (MLC) [2], and Cloonix[3] have been reviewed. Finally MLC was selected as a very light-weight emulation environment based on LinuX Containers (LXC) and the only environment which easily allows the emulation of typical wireless link characteristics like packet loss and delay and the differentiation of unicast and multicast traffic. Although MLC is an emulation framework that is still under development, it proved perfect to the requirements of this work.

1.4 Contributions of the work

The first contribution is the characterisation, in terms of packet loss and Round Trip Time (RTT), of a CN, not only statically but also dynamically. To the best of our knowledge this type of networks were unstudied from this point of view. The zone characterised has a significant number of nodes.

The second contribution is a substantial improvement of the realism of the emulated network scenarios on which the DRPs characterisation experiments are performed since our emulated scenarios are setup according to the new knowledge resulting from the first contribution. An additional contribution is the usage of a new emulation framework.

The third contribution is the evaluation of three DRPs considering a relative large number of nodes and the most accurate CN emulation we know. The previous works consider either a relative small number of nodes or highly unrealistic scenarios.

The forth contribution is the evaluation in IPv6. To the best of our knowledge all previous work is restricted to IPv4.

The fifth contribution is the inclusion of BMX6 in the set of analysed DRPs. This protocol has never appeared before in the literature.

1.5 Organization of this document

Chapter 3, Methodology, describes the aims, the design parameters and the hypotheses of the three set of experiments which were carried out. Chapter 4, Results, presents the results of each set of experiments, challenging and analysing the consequences stemming from each. Finally Chapter 5, Conclusions, Lessons Learnt and Further Work, recapitulates the outcome of this work, enumerates the lessons learnt and sketches possible lines of future research along these lines.

Appendix A, Data, sets provides information about the network under study as well as the sets of data used in the simulations. Appendix B, Interior Gateway Protocols introduces the criteria for the categorisation of the main Interior Gateway Protocols (IGPs) and classifies the analysed DRPs according to these criteria.

Chapter 1, *Introduction*, contextualises the work presented in this report, sets its aims, justifies the main choices made (DRPs, CN, metrics, etc.)and enumerates the contributions made. Chapter 2 *State of the art*, surveys the existing research prior to this work and related to the issues it deals with. It starts presenting the selected CN and discusses network details relevant to this work such as topology and the available information on link quality. Afterwards, each of the DRPs is discussed with a focus on those parameters that have influence on the overhead protocol and the convergence time. The chapter concludes with a review of the emulation framework used in the work. Chapter 3, *Methodology*, describes the aims, the design parameters and the hypotheses of the three set of experiments which were carried out. Chapter 4, *Results*, presents the results of each set of experiments, challenging and analysing the consequences stemming from each. Finally Chapter 5, *Conclusions, future work*, recapitulates the outcome of this work, enumerates the lessons learnt and sketches possible lines of future research along these lines.

Appendix A, *Data sets*,sets provides information about the network under study as well as the sets of data used in the simulations. Appendix B, *Interior Gateway Protocols*, introduces the criteria for the categorisation of the main Interior Gateway Protocols (IGPs) and classifies the analysed DRPs according to these criteria.

Chapter 2

State of the art

2.1 Community Networks

Although all Community Networks (CNs) share a traits set that characterise them as an entity [4] (universal access, promotion of the commons, etc.), in practice they vary a lot among them in terms of size, technological solutions, management, uplink distribution, etc.

Table 2.1 summarises the relevant characteristics of the CNs mentioned in this work. Some of them have been described in detail, such as Athens Wireless Metropolitan Network (AWMN) [5] or guifi.net [6].

CNs are increasingly attracting the attention of researchers of many fields. In her PhD thesis, Bina [7] studies in detail the mechanics employed for the mobilization and organization of their members based on a extensive survey. In his PhD thesis, Bona [8] inquires into the impact on politics and society of a massive generalisation of technology appropriation taking guifi.net as a case where such appropriation has fully succeeded. The economical impact of CNs and their capacity to foster the telecommunications marked has also been analysed in some specific cases [9].

Currently the European Union has to on-going research projects related to CNs: (i)

Name	Main areas	Working nodes[1]	Centralised registration?	Topology / Main DRPs	Free uplink?
guifi.net	Catalan Countries (single main cloud)	>18.000	guifi.net DB / CNML	Infrastructure / BGP + OSPF clouds	Yes (via proxies)
AWMN[2]	Attica (single main cloud)	>2.400	WiND DB	Infrastructure / OLSR + BGP clouds	No (users share uplinks privately)
Freifunk	Almost every German city[3] (isolated clouds per city)	3.000?	Somehow	MANET + P-t-P / OLSR	Yes
FunkFeuer	Viena, Graz, etc.[3] (isolated clouds per city)	330	NodeDB (each cloud independent)	MANET + P-t-P + P-t-mP / OLSR	Yes (Public IPs[5])
wlanslovenija	Ljubljana, Maribor, etc. (all clouds VPN connected)	>150	Nodewatcher	MANET + P-t-P / OLSR	Yes

[1] Node means a physical location. A node may contain several devices. As of August 2012.
[2] There are at least other 16 active CNs in Greece. Some of them have links with AWMN.
[3] Communities among cities are very different from each other.
[4] Hence, clouds are interconnected through Internet as any other network.

Table 2.1: *Summary of the CNs mentioned in this work.*

Community Networks Testbed for the Future Internet (CONFINE)[1], within the European Community Framework Programme 7 within the Future Internet Research and Experimentation Initiative (FIRE), and (ii) Commons4Europe[2], within the Competitiveness and Innovation Framework Programme of the European Union. The aims of these projects are, among others, to prove the viability of CNs as a complementary model to the existing ones to extend the existing Internet infrastructure.

2.1.1 guifi.net Community Network

guifi.net, the biggest CN in the World, is noted for its management tools. Through these applications IP assignment, routers configurations, system and network monitoring, etc. is done automatically. This high degree of automation eases the network maintenance

[1] `http://confine-project.eu/`
[2] `http://commonsforeurope.net/`

and the network expansion because any individual can become a new community member (i.e. to install his/her node) without needing any special knowledge and almost effortlessly and can start contributing to the maintenance almost since the beginning.

All guifi.net network information is stored in a database and made public via Community Network Mark Up Language (CNML)[3]. Despite the database is manually maintained through guifi.net website and is not automatically synchronised with the real network state, the information it contains matches the reality of the network almost perfectly.

As almost all CNs the wireless connections are at roof-level, meaning that specific equipment (such as Customer Premises Equipments (CPEs)) to connect is required and support for mobile devices is not foreseen.

Infrastructure is the predominant mode (i.e. dedicated links between supernodes and supernodes with Access Points (APs) giving coverage to end-users nodes) in guifi.net. An accurate channel selection and the massive usage of point-to-point[4] results in an exceptional efficient wireless network in terms of scalability, stability, bandwidth and latencies. Due to this exceptional performance it used by many members as a production network for their daily job.

guifi.net uses Border Gateway Protocol (BGP) as core routing protocol and Open Shortest Path First (OSPF) as Interior Gateway Protocol (IGP). Despite both of these routing protocols are designed for much more stable link technologies (i.e. wired networks), due to the aforementioned exceptional stability of the guifi.net network the result of such combination works fairly well.

The topology of the selected zone under study is the common one of most of guifi.net zones. The active nodes, are divided into supernodes (nodes with more than one wireless interface) and end-user nodes (nodes with a single wireless interface)[5]. Focusing on the supernodes (i.e. the nodes that actively participate in the routing decision-making process), about the half have more than one route to the other supernodes, forming a main mesh cloud, which is sometimes referred to simply as the *mesh*. While the rest

[3] Public information about the internals of the network is a must to keep the network open.

[4] Point-to-multiPoint links are considered a bad practice because the degrade the quality of the links.

[5] In infrastructure mode, and this is the case of guifi.net, to *actively* expand the network a node must have at least two wireless interfaces because multipoint links are just allowed at the edge nodes for the network performance shake. Hence the terminological distinction between supernode (those that contribute to the network expansion) and nodes (the network leaves).

form ramifications of a single path sometimes connecting smaller mesh clouds.

2.1.2 Community Network characterization

Although some CNs like guifi.net, AWMN and wlanslovenija[6] publish information about the nodes availability, none of them gives information about the relevant link parameters for the Dynamic Routing Protocols (DRPs)[7].

Despite there are some efforts in the literature to characterise other wireless networks, like in [10], to the best of our knowledge any of this information is suitable for our purposes either.

Therefore, this work will need a phase of observation of the network to be emulated in order to be able to provide realistic values of the link quality (packet loss percentage and time delay) to the emulation frame.

My Trace Route (MTR)[8] is a network diagnostic tool that combines the functionality of the traceroute and ping programs in a single application. It investigates the network connection between the host where MTR runs on and destination host by sending packets with purposely low Time To Lives (TTLs). It continues to send packets with low TTL, noting the response time of the intervening routers. This allows MTR to know the response percentage and response times of the internet route to the destination host. MTR can either work in live mode or report mode. There are many high level applications that have MTR as backend but as far as we know none of them provide the data we need.

2.2 Dynamic Routing Protocols

There have been several studies about the performance of different DRPs in wireless mesh networks.

[6] `http://wlan-si.net/`

[7] Some projects like graciasensefils or qMp give some information about the link qualities -information extracted from the DRP- but since this information is just graphically presented of no use for this work.

[8] `http://www.bitwizard.nl/mtr/`

Johnson in [11] compares the performance in terms of overhead, throughput, CPU and memory consumption of OLSR and BMXd by performing measurements on a real hardware using a 49-node indoor grid testbed.

[12] and [13] compare Optimized Link State Routing Protocol (OLSR) with Ad-Hoc On Demand Distance Vector (AODV), Destination-Sequenced Distance Vector (DSDV) and Dynamic Source Routing protocol (DSR) in terms of routing overhead, average delay and throughput; those papers study the protocols performance by means of simulation of a grid like topology and considering a portion of mobile nodes. [14] and [15] evaluate OLSR, DSDV, DSR and AODV on a real testbed; however, the number of nodes of the testbed is low (8) or even not mentioned, and scalability is not analysed.

As far as we are aware, none of the former work have investigated the performance consequences for mesh routing protocols when switching from Internet Protocol version 4 (IPv4) to Internet Protocol version 6 (IPv6).

There are many Mobile Ad-hoc NETwork (MANET) routing protocols exploring a combination of different features, [16] and [17], such as performance metrics beyond hop-count, cross-layer designs taking metrics from layer-2, scalability for large networks, robustness to mitigate service disruption due to link failures or congestion, etc.

Table 2.2 summarises the versions and other high-level implementation characteristics of the DRPs analysed in this work. Further protocol specific details are examined in the following subsections.

Protocol	**Implement.**	**Version or git revision**	**Non-stripped binary size [KB]**	**Stripped binary size [KB]**	**Modules/ plug-ins support**	**Metrics**
Babel	Babeld	1.3.3	320	83	–	ETX
OLSR	OLSRd	0.6.3	1056	377	Plug-ins	ETX
BMX6	bmx6	bf554383	2055	251	Modules	BMX6 specific

Table 2.2: *Summmary of the implementations characteristics of the DRPs analysed in this work. Binary sizes using the default Makefile.*

All analysed DRPs also have in common that:

- Routes are set-up proactively

- Link neighbors are detected automatically
- Protocol data is send to IPv6 multicast group
- User Data Protocol (UDP) is employed as Transport Layer Protocol
- Protocol messages are aggregated (i.e. a single UDP datagram may convey several protocol frames or messages).

2.2.1 Babel

Babel is a destination-sequence distance-vector routing protocol specified at RFC 6126 [18]. It is based on the Bellman-Ford protocol and uses a feasible condition to discard routes that are not guaranteed to be loop-free[9]. The Babel implementation analysed in this work is Babeld [10]. This implementation is developed and maintained by Juliusz Chroboczek who is also the author of the RFC 6126. The strategy for computing link costs and route metrics is not specified by the RFC 6126. In Babeld these computations are done using a variant of Expected Transmission Count (ETX).

Babel enjoys fairly fast convergence since it uses triggered updates and explicit requests for new routing information. It usually converges almost immediately after the link quality measure has completed. This initial solution is not optimal, after converging to a merely satisfactory set of routes, Babel slowly optimises the routing tables. In the presence of heavy packet loss, converging on an optimal set of routes may take significantly longer since triggered route updates can get lost and are only recovered by a following mandatory periodic route update which is send rather seldomly (with a default interval of 20 seconds).

In order to decrease the protocol overhead Babel allows to omit subnet prefixes when multiple addresses are sent in a single packet as described in [19]

[9] According to RFC 6126 “Babel is a mostly loop-free distance vector protocol”. RFC 6126 describes the situations where loop-freedom cannot be granted.

[10] `http://http://www.pps.univ-paris-diderot.fr/~jch/software/babel/`

2.2.2 OLSR

Optimized Link State Routing Protocol (OLSR), as specified in the RFC 3626 [20], is a proactive routing protocol that uses an optimised version of a pure link-state protocol. It is optimised in terms of overhead, since topology control messages are not purely flooded through the network, but selectively by the MultiPoint Relays (MPR). MPRs are selected in a distributed fashion, so each node selects a small set of immediate neighbours to be its set of MPR, which satisfy that every 2-hop away neighbour can be reached through one of the nodes on the MPR set.

However, its wide usage in existing CNs has shown that the MPR based optimisation is inefficient when faced with the dynamic changes and poor links that occur in real-life and self managed deployments. To overcome this, the MPR algorithm is disabled in Optimized Link State Routing Protocol daemon (OLSRd) [11], the currently most used OLSR implementation and the analysed in this work. In OLSRd the fish-eye extension is activated by default to reduce the average protocol traffic overhead and to enhance scalability, [21] and [22], and the Hop Count metric of the RFC has been replaced by ETX. Due to all these and other changes OLSRd became RFC-incompatible since a long time.

Currently, most existing CNs are using OLSRd implementation for the whole network (e.g. FreiFunk[12], FunkFeuer[13]) or in parts of the network (e.g. guifi.net[14], AWMN[15]). Since its first larger deployments in community networks in 2003, the code has constantly improved and become a very stable, mature, and future rich solution for small and large-scale mesh projects.

OLSR has been described, analysed, and discussed extensively during previous work, [23] and [24]. In the following we are just briefly reviewing the most important principles and messages of the OLSR implementation used for our evaluations and how they relate to protocol traffic overhead and convergence time.

OLSR periodically broadcasts two types of messages:

[11] http://www.olsr.org/
[12] http://start.freifunk.net/
[13] http://funkfeuer.at/
[14] http://guifi.net/
[15] http://www.awmn.gr/

- HELLO messages are broadcast every 2 seconds by default by every node and only travel one hop. HELLO messages mainly contain the sender's IP, a list of its neighbours, and the link status. They are used to calculate the link qualities between nodes.

- Topology Control (TC) messages are flooded through all the network. In case of disabled MPR algorithm, these messages are originated by all nodes (otherwise TC messages are flooded selectively by the nodes that are selected as MPRs). TC messages have an originator address and a list of its neighbours with corresponding link qualities. TC messages are processed by each node to internally calculate the full topology graph of the network which provides the basis for calculating the best next hop to any given destination.

Like any link-state routing protocol, OLSR is conceptually vulnerable to routing loops resulting from non-synchronised topology graphs as calculated by different nodes on the forwarding path of a data packet. A trade off to this problem is given by flooding TC messages at a smaller interval, allowing nodes to recalculate their topology view more often at the cost of increased protocol traffic overhead and CPU load. The fish-eye extension for OLSR implements a third way to mitigate the problem. It is based on the finding that routing loops usually occur between nearby nodes (thus nodes at one or two hop distance). To achieve better synchronisation of topology graphs between nearby nodes while allowing less frequent synchronisation between distant nodes, TC messages are flooded with different TTL values. Specifically, the sequence of TTLs with active Fish-eye extension in the OLSR implementation used for our evaluations is 2,8,2,16,2,8,2,255. This means that only every even TC message is flooded beyond its two-hop neighbourhood. Since by default the activation of the fish-eye extension is delayed 140 seconds a transient state with higher protocol overhead must be expected. This delayed activation is meant to reduce the convergence time after booting a node.

2.2.3 BMX6

BatMan-eXperimental version 6 (BMX6)[16] is the successor of the BatMan-eXperimental daemon (BMXd) which emerged as an independent branch from the BATMAN routing

[16] `http://www.bmx6.net/`

protocol [25] to explore and test new approaches for routing and context dissemination in mesh networks. The design and development of this new version was driven by the objective to better cope with the increased address space given by IPv6 addresses, enable node-individual configurations while clarifying the handling of conflicting node announcements (e.g. duplicate address allocations), and allow efficient state dissemination (thus reduced protocol overhead) through the strict distinction between local and global as well as static and dynamic state.

BMX6 as well as BMXd are actively used in current CNs and projects such as guifi.net qMp[17] and Graciasensefils[18], Freifunk, and Lugro-mesh[19].

BMX6 is a table-driven routing protocol for wireless mesh networks. As any table-driven routing protocol, its goal is to compose a path from source to destination by deciding on each node which will be the next hop. BMX6 is a distance-vector protocol, since the information each node manages is a list of tuples of nodes' identifiers and the cost of getting there when choosing a concrete link: <destination node, next hop, cost>. The novelty in BMX6 is the dissemination mechanism it uses to propagate this information. The dissemination protocol is inspired by human social networks that are scalable as people tend to learn more about its neighbourhood and abstract and filter out information about others. Topology knowledge in a node is optimised for itself and its neighbours by using local compact identifiers for a local compressed stateful dialogue.

During the transient phase, neighbours exchange knowledge about their environment: nodes' descriptions, links, etc. and provide information about their Individual IDentifier s (IIDs), which identify nodes in a compact way. With this information, each node sets up a dictionary table per neighbour that translates its IIDs values to the globally unique and non-ambiguous hashes of the full node description. On the steady state, each node has a local information state in the form of IID-to-hash dictionaries; and a global information state as hash-to-description dictionary. During this phase the protocol just exchanges small packets to keep track on the variation of link metrics and to monitor network changes. Thanks to the information state deployed during the transient phase, the fields of this periodically exchanged routing updates, which are usually given by a 128 bit IPv6 address, can be substituted by the much shorter IID value (16 bits), and thus it results in compressed messages. The separation in local and global state also

[17] http://qmp.cat/
[18] http://graciasensefils.net/
[19] http://www.lugro-mesh.org.ar/

pays off when a node moves, and therefore, its neighbourhood changes, because it only needs to re-establishment of IID-to-hash relations, whereas already existing knowledge about pairs between hashes and corresponding descriptions is still valid.

As a result, the control overhead increases when there is a network change, stabilising afterwards to a lower value. Hence, when a node boots we must expect a considerable peak at the very beginning, result from the exchange of nodes descriptions and local IID tables. However, after the initial transient phase is over, future network changes -connectivity variations- will have much smaller impact on the traffic overhead because just very little of the information already exchanged during the initial transient phase must be updated -the one referring to the current change.

It needs to be considered, that a typical situation would be small changes in the network, like connecting or disconnecting a node from the network, while simultaneous booting of all the nodes in the network is not common. However, a similar effect can be expected in case that two separate mesh clouds become one single network by the deployment of a new link that interconnects them. On this case, we can expect a high peak of traffic, since every node on the network needs to learn about all the nodes in the other cloud.

Consequently, there are two different types of messages on BMX6 depending on their nature: (i) periodic messages, that are periodically generated by the protocol on every node; and (ii) occasional messages, that are exchanged only when necessary because of a change in the network.

The periodic messages generated by BMX6 are responsible for the little overhead during the steady phase, and they are:

- Hello advertisement (HELLO_ADV) messages are broadcasted every HELLO_-INTERVAL, which by default is 0.5 seconds. They are used to measure the link quality (based on the number of received messages) and to know whether a link is alive or not.

- Similarly, report advertisement (RP_ADV) messages are periodically broadcasted as response to the HELLO_ADV messages, and therefore every HELLO_INTER-VAL. They provide a summary of the received and lost hello messages from all neighbours and related links.

- OGM_ADVs or OriGinator Messages are sent every OGM_INTERVAL (which by default is 5 seconds) and propagated over the network. They are used to let nodes become aware of other nodes further than just one hop away and inform about the path metric to the originating node. However OGM_ADVs are not flooded indiscriminately through the network, but just through so-called relevant links. A link is relevant whenever it is necessary to reach one of the nodes in the network, i.e. it is the next hop of at least one entry in the routing table.

In contrast, the occasional messages create a peak of traffic when there is a change on the network, allowing nodes to gain knowledge about their neighbourhood or learn about the full description of a formerly unknown node. These messages are:

- Link advertisement (LINK_ADV) and optional device advertisement (DEV_ADV) messages are broadcasted on demand (due to the reception of LINK_REQ or DEV_REQ messages) to describe the existence and further attributes of network devices and links from the perspective of an individual node. Each LINK_ADV message represents a link as perceived (due to previous received HELLO_ADVs) by the transmitting node to one of each neighbours. The order in which LINK_-ADV messages are aggregated is further used as an implicit reference to a specific link of the node when creating or processing RP_ADVs messages.
- Description advertisement (DESC_ADV) messages are exchanged between nodes, providing a full description of a specific node, containing details such as their IP addresses, hostname, and protocol parameters. Description messages are requested via DESC_REQs messages due to the receipt of an unknown description hash.
- A hash advertisement (HASH_ADV) message provides the relation of a node-specific IID value to the hash of a specific node description that is used for globally non-ambiguous node identification. By means of description's hashes BMX6 refers to already known nodes without having to send the full description of the node. HASH_ADV messages are requested whenever an unknown IID reference or a message from an unknown node is received.

In summary, BMX6 achieves to reduce its overhead by using two different mechanisms: first, it optimises the traffic transmitted periodically through the network by means of

establishing a common understanding between neighbours using compact IIDs and description hashes; secondly, it controls the flooding of messages by analysing whether a link is relevant or not, and omits non-relevant links on the flooding of OGMs.

2.3 Networks emulation

2.3.1 Contextualisation

Contextualisation, also known as operating-system level virtualisation or simply as containers, do not run virtual machines at all, but simply segregate multiple user space environments from each other, while everything runs under one kernel. Therefore, containers are not virtualization technologies per se but carve up a single system in *super chroot* jails[20]. All the *guest* processes in the containers run directly on the same *host* kernel and as such, generally have access to the same CPU, RAM, etc. resources as the host.

Hence, contextualisation do not allow to run multiple different kernels, but allow different root filesystems in the different containers. Container systems tend to have low overhead and high density, but also lower isolation between the different containers. While possibly limiting for testing or development, they can simplify production usage since the shared kernel reduces the amount of software and security maintenance.

There are currently three main implementation of containers for the Linux kernel that are free software: OpenVZ[21], Linux-VServer[22] and LinuX Containers (LXC)[23]. OpenVZ consists of a modified Linux kernel that provides virtualisation/isolation, resource management, and checkpointing and some user-level tools. Linux-VServer also consists of a a modified Linux kernel and its user-level tools but does not provide checkpointing and just provides partially network isolation. LXC is in the Linux kernel mainstream. It provides resource management and network isolation but not checkpointing. Parallels Virtuozzo Containers is a proprietary contextualisation project for the Linux Kernel that can change memory and CPU quota during runtime.

[20] Chroot jails are the precursors of containers.
[21] `http://openvz.org/`
[22] `http://linux-vserver.org/`
[23] `http://lxc.sourceforge.net/`

In [26] OpenVZ represents contextualisation in a comparison of containers against para-virtualisation and full-virtualisation. Linux-VServer analised in detail in [27]. To the best of our knowledge no performance evaluation of LXC has appeared in the literature yet.

2.3.2 Network emulators

Network emulators are applications aimed at helping users in the management of virtual machines and of the underlying network connecting those machines. Using contextualisation as the virtualisation technique provides clear benefits in terms of scalability of number of nodes. Mininet[24], well described in [28], makes usage of LXC. Cloonix[25] also supports LXC, among others, but this support is not granted in the near future.

2.3.3 Mesh Linux Containers

Mesh Linux Containers (MLC)[26] is a collection of scripts based on LXC and Linux networking tools. MLC's goal is to provide the necessary tools and scripts to quickly create emulated network topologies including link-specific packet loss and delay with up to hundreds of nodes.

Using MLC on a single testbed machine (a 2.4GHz Pentium I5 with 4 cores and 4GB of RAM) we could emulate up to 200 nodes running either OLSR or BMX6 in its default configuration and connected in a grid-like topology with perfect links before the system got overloaded. With 200 nodes the CPU load (as measured with top) exceeded 75 percent only during the protocol startup phases.

MLC creates a base LXC container with all the necessary software that will be run on the testbed. Afterwards, this container is replicated with the proper routing configuration files and network configuration depending on the desired number of emulated nodes,

[24] `http://yuba.stanford.edu/foswiki/bin/view/OpenFlow/Mininet`
[25] `http://clownix.net/`
[26] `https://github.com/axn/mlc`

On the network side, each container has 3 interfaces which connect with the other containers through 3 different bridges. The first bridge is intended for controlling the containers, while the other two can be used for experimentation. MLC allows the creation of virtual links between the containers on the last two interfaces by controlling the forwarding probability and delay at link level. Within these restrictions it allows the definition of any target network topology. For emulating different link characteristics MLC defines different levels of link qualities in terms of delay and error probability and depending on unicast or broadcast traffic.

Chapter 3

Methodology

After an initial set of experiments aimed to gather information of the real network to feed the emulation framework with realistic scenarios two other subsets of experiment follows. In order to clearly decouple the affect of variation of the number of nodes and links from the influence of the link quality dynamics, the link qualities of the second subset of experiments is kept constant while only the number of nodes and links varies. Given this static nature of the links the scenarios of this experiments set is also referenced as *static scenarios*. On the contrary, in the third subset of experiments while both, the number of nodes and links remain constant, being the link quality of these links vary according to the results of the results of the initial set of experiments. The scenario of this experiments set is also referenced as *dynamic scenario*.

3.1 Community Network characterisation

3.1.1 Aim

The purpose of this set of experiments is to characterise the links of the selected network in terms of packet loss and round trip over time.

3.1.2 Methodology

My Trace Route (MTR) has been run in short periods and several times per destination node. All the tests have been run in the same server which is connected to a mesh node[1].

In order to avoid link saturation and the resulting distortion in the measurements, on the one hand the destination nodes tested at the same time have been divided in subsets being a single subset test at a time using the round robin scheduling algorithm, and on the other hand the frequency of the probes has deliberately been pitched low. In compensation destination nodes have carefully distributed among subsets in order to maximise the links covered by each subset. Additionally, whereas in the static characterisation the all nodes have been probed aiming to reach the maximum possible number of links, during the dynamic characterisation tests the number of probed nodes has been slightly reduced by discarding those which paths were very likely covered by other probes because in this case the information update frequency had prevalence over marginal links consideration.

Regarding the network topology and initial graph has been built processing the guifi.net database information of the selected zone, Barcelona. To do so just the nodes and links registered as *working* have been considered. To this initial graph we manually included those nodes and links from the adjacent zones, Badalona and Hospitalet del Llobregat, that we already knew that were in the routes to some of nodes we were already considering[2]. An initial set of MTR experiments allowed us to identify and eliminate those nodes that despite of being registered as working in the guifi.net database showed to be down[3]. Theses tests also helped us to identify and incorporate to the graph links that were not registered in the database. Finally we resolved most of the few remaining topology opened questions, like links that stopped working do to the increase of interferences of or specific customisations, contacting the administrators/owners of the affected nodes. From now on the resulting graph of this process may be referred to as *main*/*initial* graph/network.

[1] In this work the node identifiers (nodeIDs) are the same as in the guifi.net database. Community Network Mark Up Language (CNML) description of each node is accessible via http://guifi.net/ca/guifi/cnml/<nodeID>/. Figure 4.1 shows nodeIDs and the links.

[2] Geopolitical divisions, on which guifi.net zones are based, may not match network zones, specially in such dense areas as Barcelonès.

[3] Our results were contrasted with the information of the guifi.net website where every device is monitored for a year.

3.1.3 Experiments design

The results of two campaign of measurements are explicitly used in this work. The first to give an overview of the link quality dynamics and to identify any possible behaviour patterns. The second is meant to provide information of the link quality dynamics. The settings of the preliminary results to determine the initial graph has been omitted.

Community Network characterisation **One week long measurements**	
Start	Wed, 22 Aug 2012 14:05:23 GMT
End	Wed, 29 Aug 2012 14:04:45 GMT
Duration	One week
Source nodeID	7281
Destination nodes	66
Nodes selection	All nodes
Nodes per subset	10
Cycles per iteration	60
Cycles interval [s]	1.0
Packet size [bytes]	1400

Table 3.1: *One-week-long measurement campaign settings.*

Community Network characterisation **One hour long measurements**	
Start	Thu, 30 Aug 2012 14:43:41 GMT
End	Thu, 30 Aug 2012 15:44:35 GMT
Duration	One hour
Source nodeID	7281
Destination nodes	50
Nodes selection	Leaves + mesh nodes likely not to be in the path of any leaf
Nodes per subset	10
Cycles per iteration	40
Cycles interval [s]	0.5
Packet size [bytes]	1400

Table 3.2: *One hour long measurement campaign settings.*

3.1.4 Hypotheses

This set of experiments has been done under the following premises and known limitations:

Symmetric links

Wireless link properties must not necessarily be symmetric. Measuring link qualities in each direction would imply being given access to each edge of each link, hence to all nodes, to make measurements. In a network were antennas are accurately aligned such as guifi.net symmetry is a reasonable assumption.

Measured unicast link properties assimilated for emulation of broadcast packet loss

Whereas in wired networks, multicast and unicast frames are transmitted identically on the the Media Acces Control (MAC) layer, there are significant differences between unicast and multicast frame transmissions in 802.11. In order to cope with the higher frame loss and collision rates in the wireless networks the 802.11 MAC protocol[29] mandates acknowledgements of received unicast frames and retransmissions of non-acknowledged frames. In contrast, multicast traffic is not acknowledged and thus never retransmitted on the MAC. Therefore, the loss ratios as seen on the IP-layer are higher than for unicast traffic. Furthermore, the 802.11 offers the distributed coordination function (DCF) mechanism [4] to protect unicast traffic from interference loss due to two simultaneous transmission attempts by stations that cannot sense each other (the hidden-station problem). Such mechanism is not allow for multicast traffic.

On the other hand, in order to mitigate the lack of acknowledgments and DCF function for multicast transmissions, packets are by default transmitted with the most robust transmission rate.

Since protocol traffic of all evaluated DRPs is send as broadcast traffic, in theory, only the broadcast link-quality characteristics are relevant for the resulting convergence performance. Unfortunately, measureing broadcast link characteristics of many links in the selected CN area is only possible with direct link-layer access to all involved nodes. This was not possible during the performed measurement

[4] The Distributed Coordination Function (DCF). Based on an RTS/CTS (request to send / clear to send) frame exchange sequence.

campaigns. Instead, measured unicast link characteristics were used for the emulation which could be rather easily deduced from mtr output by tracing unicast packet losses over a multihop link from a few source nodes to many distant destination node in the zone. As a consequence, the broadcast packet loss used for our emulation shall be considered as a best case, knowing that even worse loss characteristics may be reality.

Links not characterised

Few links of the network could not be characterised because they could not be reached from the measurement node due to the presence of alternative routes (e.g. 38971-38970). These links have been characterised as perfect links.

Results cannot be extrapolated to Ad-Hoc networks

Since the network characterised is a point-to-point network the results obtained cannot be extrapolated to Ad-Hoc mesh networks. However both the tools and the methodology developed can be applied to these networks.

3.2 Dynamic Routing Protocols performance measurement

3.2.1 Aim

The purpose of this set of experiments is to characterise how number of nodes and links affect convergence time and protocol overhead and afterwards to check if the addition of the variation of quality of links over time affect these metrics.

3.2.2 Methodology

The initial experiment of this set consisted of starting all the Dynamic Routing Protocol (DRP) deamons at the same time considering all nodes and based on link quality averages of the one hour campaign. Starting all routing deamons at once is a very unlikely situation to happen in reality but it is kind of borderline experiment that provides very intuitive information about the DRP performance.

The lack of realism in the daemons starting process is resolved since the second experiment. Henceforth, daemons are started one after another with a random time in-between. Apart form adding realism this new starting policy eliminates any potential time dependency of the results[5]. This second experiments is aimed to show the effect of this change in the starting process. Additionally, a new node is attached to the network in the steady state to see how DRPs react.

The set of the following experiments is intended to provide information about how the DRPs performance is affected by the of number of nodes. In this case the DRPs are executed several times on different networks that are subsets of the initial network. The manner these networks are fashioned ensures that all nodes belong to a single cloud. The protocol overhead is evaluated comparing the average of the protocol overhead of each DRPs run on each network for the same period of time in the steady state. Convergence time is evaluated comparing the time a new node takes to reach the farthest node (in terms of hops). All the experiments are repeated twice, one time using the package loss average resulting of the one-hour-long measurement campaign (static scenarios) and another time using the dynamic data set (dynamic scenarios).

The last set of experiments is aimed to study the convergence time of the DRP. I has been done by identifying the longest path of each of the networks considered in the previous set of experiments, attaching a new node as a neighbour to one one of its edges and measuring the time the other edge takes to become aware and succesfully route packets with the new node. Additionally the experiment has been repeated on the initial network for those number of hops that had not been covered by the initial subset of experiments. All experiments have been repeated 20 times for the static and the dynamic scenarios.

3.2.3 Experiments design

The two first experiments consider the same scenario (all nodes, static configuration) being the DRP activation the only change. This can be noticed comparing Table 3.3 with Table 3.4. Whereas in Table 3.3 all periods are exactly defined, in Table 3.4 the *rise up time* parameter is introduced. The rise up time is the time taken to start all the nodes. Since each new protocol activation is delayed by a random time between 0 and

[5] A given specific start sequence may favour a DRP.

10s, the overall rise up time varies between 0s and 660s. And because each consecutive delay is recalculated on demand the exact duration of each experiment is not predefined.

DRP overhead characterisation **Daemons started in parallel.**	
Iterations	1
Number of nodes	66
Nodes selection	All nodes
Time sequence	0s, all deamons started rise up time + 300s, daemon of the new node started rise up time + 360s, experiment finished
Link qualities	Average (1h campaign)
Duration	360s

Table 3.3: *DRPs protocol overhead experiment settings. All nodes, all daemons started at the same time.*

DRP overhead time characterisation **Daemons started in serial**	
Iterations	1
Number of nodes	66
Nodes selection	All nodes
Daemons start	Serial Delay time in-between random uniform [0,10s]
Time sequence	0s, daemon of first node started rise up time, daemon of the last node started rise up time + 300s, daemon of the new node started rise up time + 360s, experiment finished
Link qualities	Average (1h campaign)
Duration	rise up time + 360s

Table 3.4: *DRPs protocol overhead experiment settings. All nodes, all daemons started in serial.*

Table 3.5 shows the parameters of the experiments subset aimed to characterise the protocol overhead. In these experiments the main network is sampled as follows:

1. *Node 18213 is chosen from the initial graph and added to the result graph.*
2. *Among all the neighbours of this node, another one is randomly chosen and added to the result graph.*

3. *The process is repeated, choosing one node randomly among the neighbours of the previously selected nodes until the network has the desired size.*

4. *All the existing links among the selected nodes are also added to the resulting graph.*

DRP overhead characterisation **Daemons started in serial. Integration period: 1hour**	
Iterations	20/netowrk (140 in total)
Number of nodes per network	10, 20, 30, 40, 50, 60, 66
Nodes selection	By 10^{th} according to the selection algorithm
Daemons start	Serial Delay time in-between random uniform [0,10s]
Time sequence	0s, daemon of first node started rise up time, daemon of the last node started rise up time + 300s, integration period started rise up time + 3900s, integration period finished rise up time + 3900s, daemon of the new node started[1] rise up time + 3960s, experiment finished
Static experiments link qualities	Average (1h measurements campaign)
Dynamic experiments link qualities	1h in steps of 20s, 40s, 60s, 80s or 100s] (1h measurements campaign)
Duration	rise time + 3960s[1]

[1] Convergence time experiments of the sampled networks are performed immediately after overhead experiments are finished.

Table 3.5: *DRPs protocol overhead experiments settings.*

Some of the convergence time experiments are performed taking advantage of the overhead experiments set-up as described in 3.5. The remaining experiments are performed according to Table 3.6.

Convergence time is estimated by subtracting the time of the fist valid ping to the farthest node from the time of the first valid ping to the neighbouring node and as shown in figure 3.1.

DRP convergence time characterisation **Daemons started in serial. Integration period: 1h**	
Iterations	20/hop test (140 in total)
Number hops per test per network	4, 5, 6, 7, 8, 9, 10, 11
Destination node selection	Increasing by 1 the number of hops each time
Daemons start	Serial Delay time in-between random uniform [0,10s]
Time sequence of hop tests[1]	0s, daemon of first node started rise up time, daemon of the last node started rise up time + 300s, integration period started rise up time + 3900s, integration period finished
Static experiments link qualities	Average (1h measurements campaign)
Dynamic experiments link qualities	1h in steps of 20s, 40s, 60s, 80s or 100s] (1h measurements campaign)
Duration	rise time + 360s

[1] Convergence time experiments of the sampled networks are performed immediately after overhead experiments are finished as specified in Table 3.5.

Table 3.6: *DRPs convergence time experiments settings.*

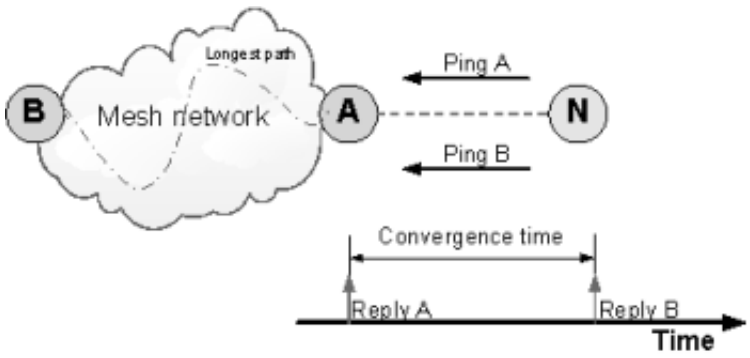

Figure 3.1: *Convergence time measurement method.*

3.2.4 Hypotheses

Apart from *symmetric links* and *multicast/unicast* assumptions explained in the previous section this set of experiments has been done under the following premises and known limitations:

Implementations default configuration

All emulated DRPs are used in their default configuration as provided by the cor-

responding implementation version (see Table 2.2 for details). Apart from the interface and IPv6 mode selection, no additional configuration hase been applied. Especially, despite Optimized Link State Routing Protocol daemon (OLSRd) and BatMan-eXperimental version 6 (BMX6) have plenty of configuration options all performance tests have been done using the daemons default configuration in order to preserve the experiment neutrality.

Discrete Link Quality emulation values

The emulation framework allows the configuration of discrete link qualities by assigning each emulated link with one out of seven possible link quality classes. These link-quality classes have been configured to cover the full range of observed link qualities from our measurement at the cost of lost resolution accuracy.

Protocol performance is only affected by broadcast link characterics (as explained in Section 3.1.4).

For broadcast traffic the following discrete link qualities have been used (in terms of packet loss in percent) from best toworst: 0, 2, 5, 10, 20, 40, 80.

On the other hand, the convergence performance of studied DRPs is measured based on probing the end-to-end connectivity over pathes by sending ping requests and tracking the first successful reception of a corresponding ping reply (as described in Section 3.2.3). Since icmp ping requests and replies are send as unicast packets and we do not want the ping observation to be affected by randomized packet losses, all unicast link characteristics are configured to zero loss. The full table of applied link classes is given in Appendix in Table A.2.

Chapter 4

Results

4.1 Community Network characterisation

4.1.1 Main graph

As of August 2012 we have counted a total of 213 active nodes in the Barcelona Guifi.net zone. These node can be divided into 66 supernodes and 153 end-user nodes. The main mesh cloud has 25 supernodes. 13 of he remaining are *daisy-chained* and the remaining ones are leave nodes. The total number of backbone links (i.e. links connecting supernodes) counted is 69, 25 of them forming the core mesh.

Figure 4.1 shows the resulting initial graph. Henceforth, NodeIDs are omitted in the figures for the sake of clarity.

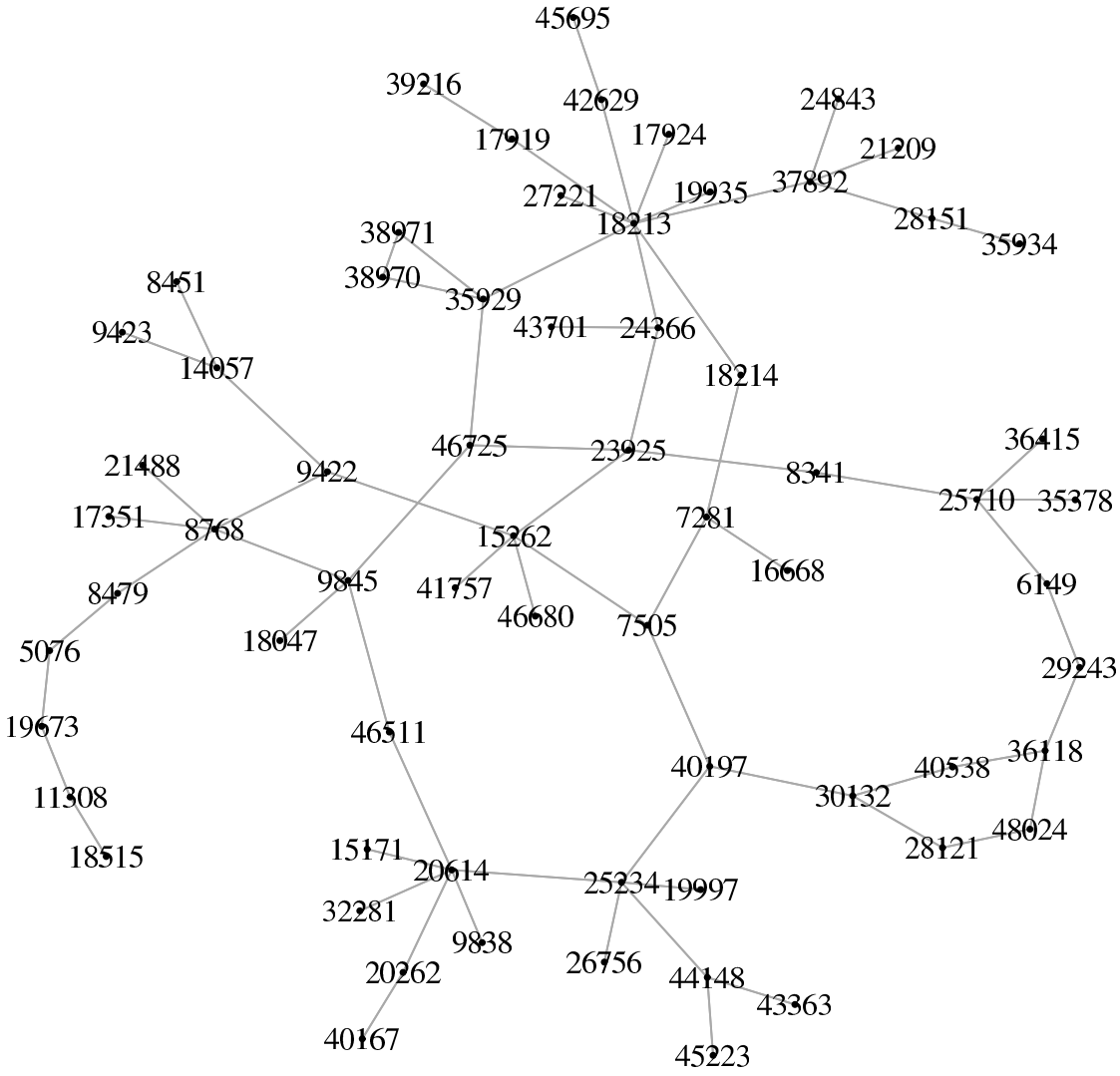

Figure 4.1: *The main network graph with nodeIDs.*

4.1.2 One-week-long measurement campaign

Despite the general fine tuning of the core links, routing changes occasionally happen. Therefore the longer the measurement period the higher the probability to gather information of not initially accessible links. In this measurement campaign just 3 links (6149-29243, 36118-48024, 38970-38971) could not be covered because no end-to-end route crossing these links from the accessible My Trace Route (MTR) source could be found. Also, as a consequence of occurred routing changes, information of 12 links, that do not belong to the main network, have been gathered but discarded for the evaluation.

The acquired data indicates that link qualities (i.e. packet losses and Round Trip Times (RTTs)) are time independent suggesting that under normal conditions they might follow a normal distribution under normal conditions as depicted in Figure 4.2 and Figure 4.3.

IEEE 802.11 is interference-prone. Right plots of the aforementioned figures show the effect on the characterisation metrics of links exposed to an interference period. Taking into account that the two links depicted belong to the same node we can additionally state that in this case the whole node was affected[1]. According to the results during the interference periods the links suffer of such degradation that they become totally useless. As a consequence any Dynamic Routing Protocol (DRP) should temporary avoid such links if possible.

The fact that even the good links seldom suffer from sporadic link degradation suggest the presence of bufferbloats[30]. Nevertheless the clarification of this point would require further analysis including the performance of specific experiments specially focusing on the Transmission Control Protocol (TCP) performance.

It should be noted that the main objective of the MTR probes is limited to collect time based data about link qualities to be used in the succeeding experiments. Although, modelling the link quality is out of the scope of this work, as a result of our analysis, we can say that a model should include at least the following items:

Links in normal conditions According to our results links can be categorised (e.g. excellent, good, bad and poor). A normal distribution may be enough to characterise

[1] Whereas all the six links that node 20614 has followed the same pattern the remaining links its neighbours had a normal behaviour

each category. The results here presented might be enough for an acceptable first approximation (See Figure 4.4).

Interferences Middle to long periods of extreme high metrics. A good approximation would be to drop most of the packets and apply a high delay to the remaining ones. In the absence of specific information the probability of an interference happening periods and its length can be modelled by a semi-infinite interval distribution function

Bufferbloats Short to middle-short period of extreme high metrics. In the absence of further information can be modelled by the same distribution functions of interferences.

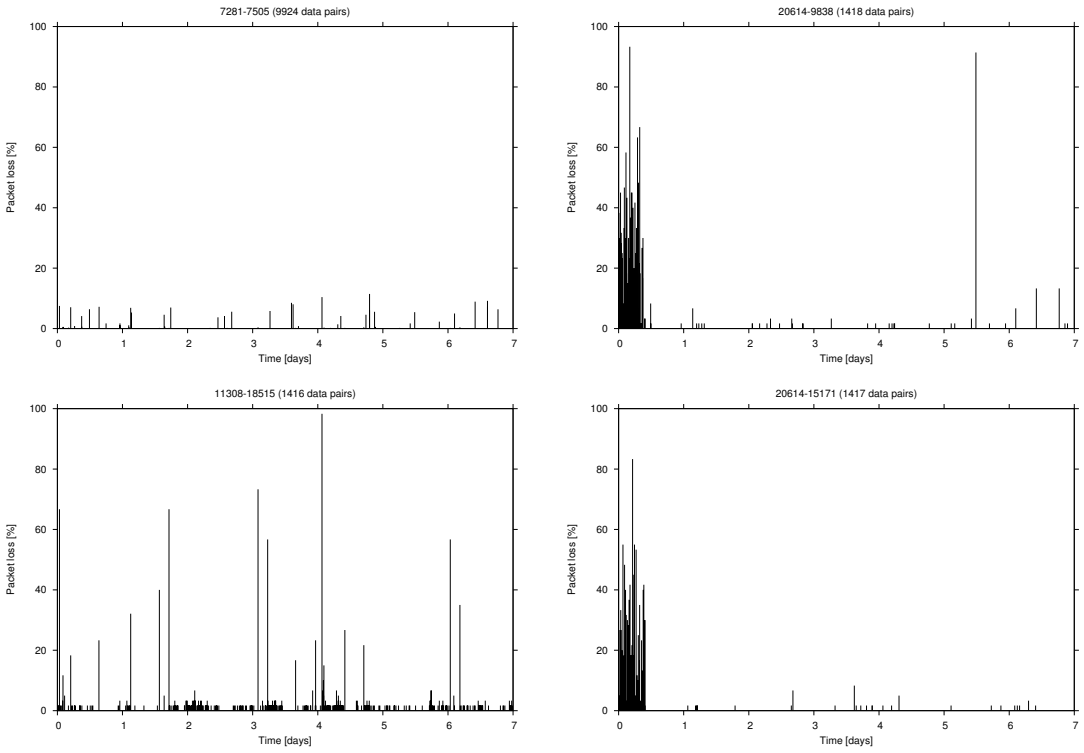

Figure 4.2: *Packet losses of an excellent link (top left), a poor link (bottom left) and two links of a node affected by interferences during the first half measurement day (top right and bottom right). Parametrisation in Table 3.1.*

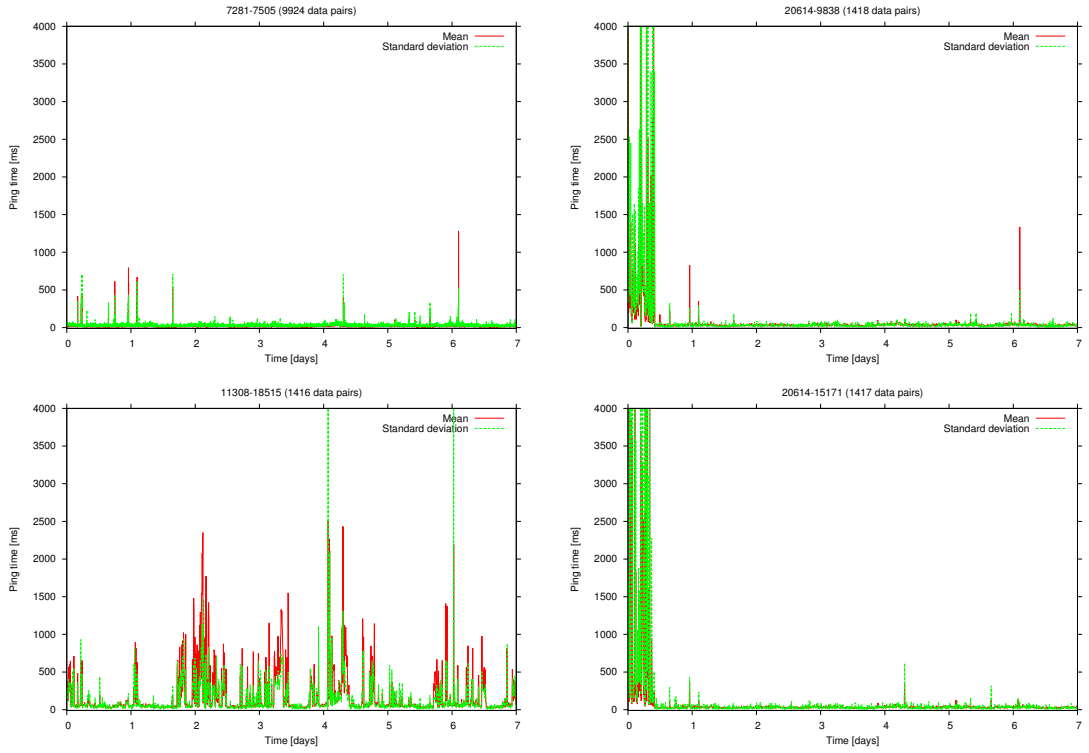

Figure 4.3: *RTTs of an excellent link (top left), a poor link (bottom left) and two links of node affected by interferences during the first half measurement day (top right and bottom right). Parametrisation in to Table 3.1.*

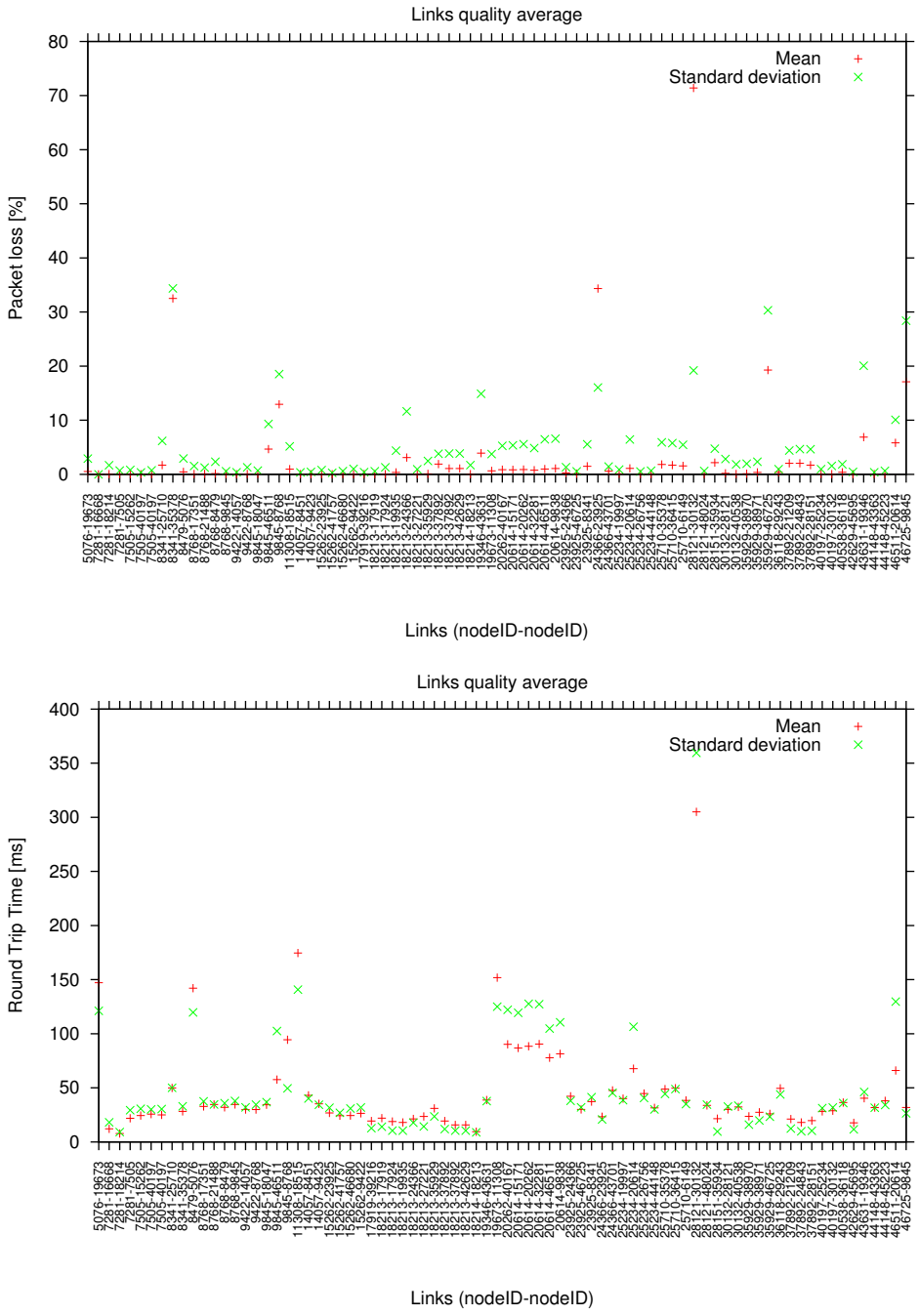

Figure 4.4: *Average packet loss (top) and RTTs (bottom) of all links of one week long measurement campaign. Parametrisation in Table 3.1.*

4.1.3 One-hour-long measurement campaign

In this measurement campaign 6 links (6149-29243, 9845-46511, 18213-25366, 35929-18213, 36118-48024, 38970-38971) have remained untested and information of 6 links that do not belong to the main network has been gathered.

Before any further analysis it has to be emphasized that lack of information of some of the inner mesh links precludes any fair comparison between the performance of the real network and the results here presented. To solve this issue other points of measurement would be necessary in order to acquire the data required to characterise these links.

In contrast to what one might initially think, links colouration of Figure 4.5 (mean and standard deviation of each link) must not necessarily exactly correlate with the colouration of Figure 4.6, (emulated changes per link), because the range of the link quality steps of the emulator growths nearly exponentially (See Table A.2). Indeed, the worse the metrics are the less relevant how they change since the link should only be selected if there is not any other alternative route, in which case link quality becomes irrelevant due to the lack of alternatives.

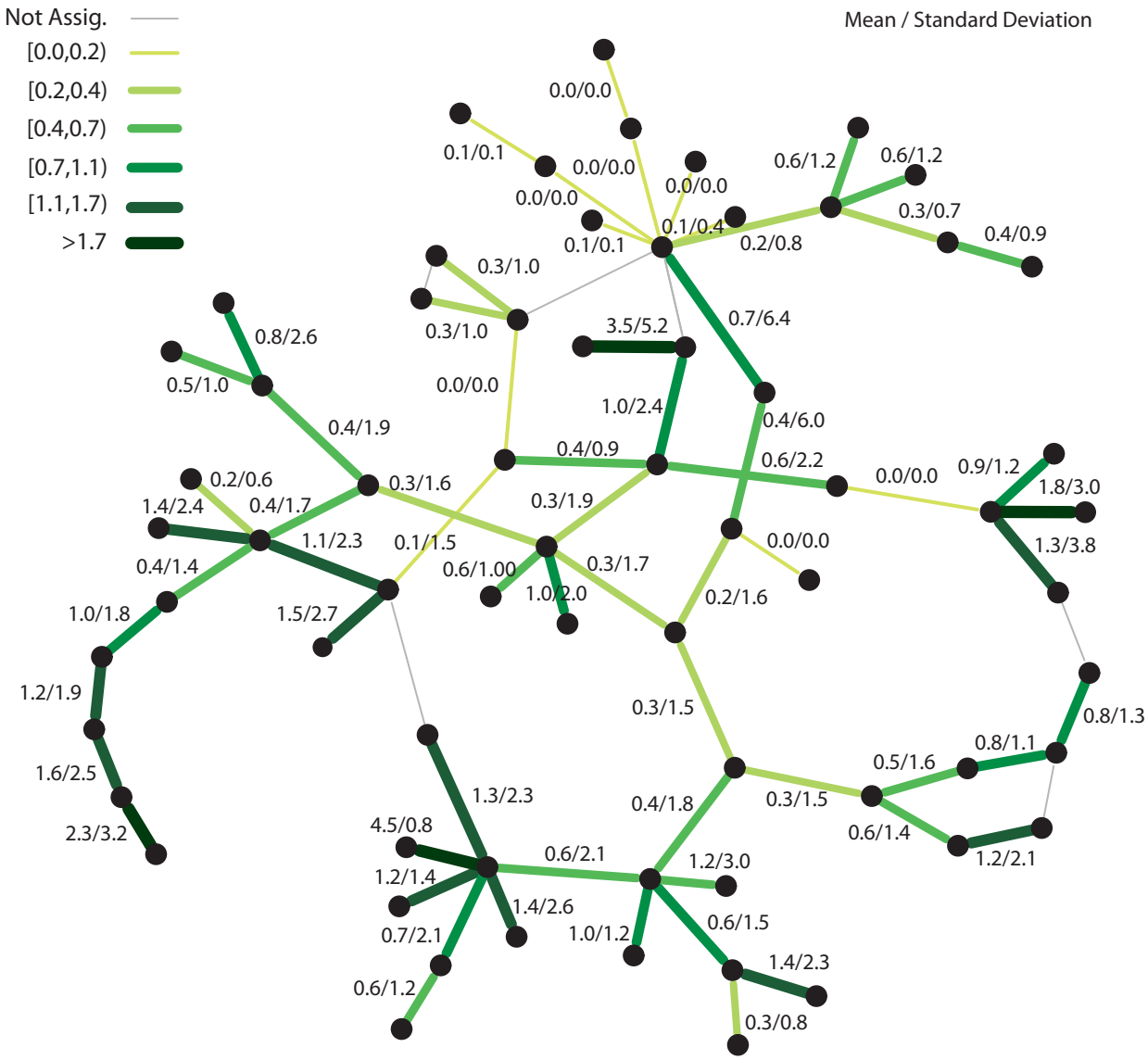

Figure 4.5: *Mean and standard deviation of link quality per hour. Parametrisation in Table 3.2.*

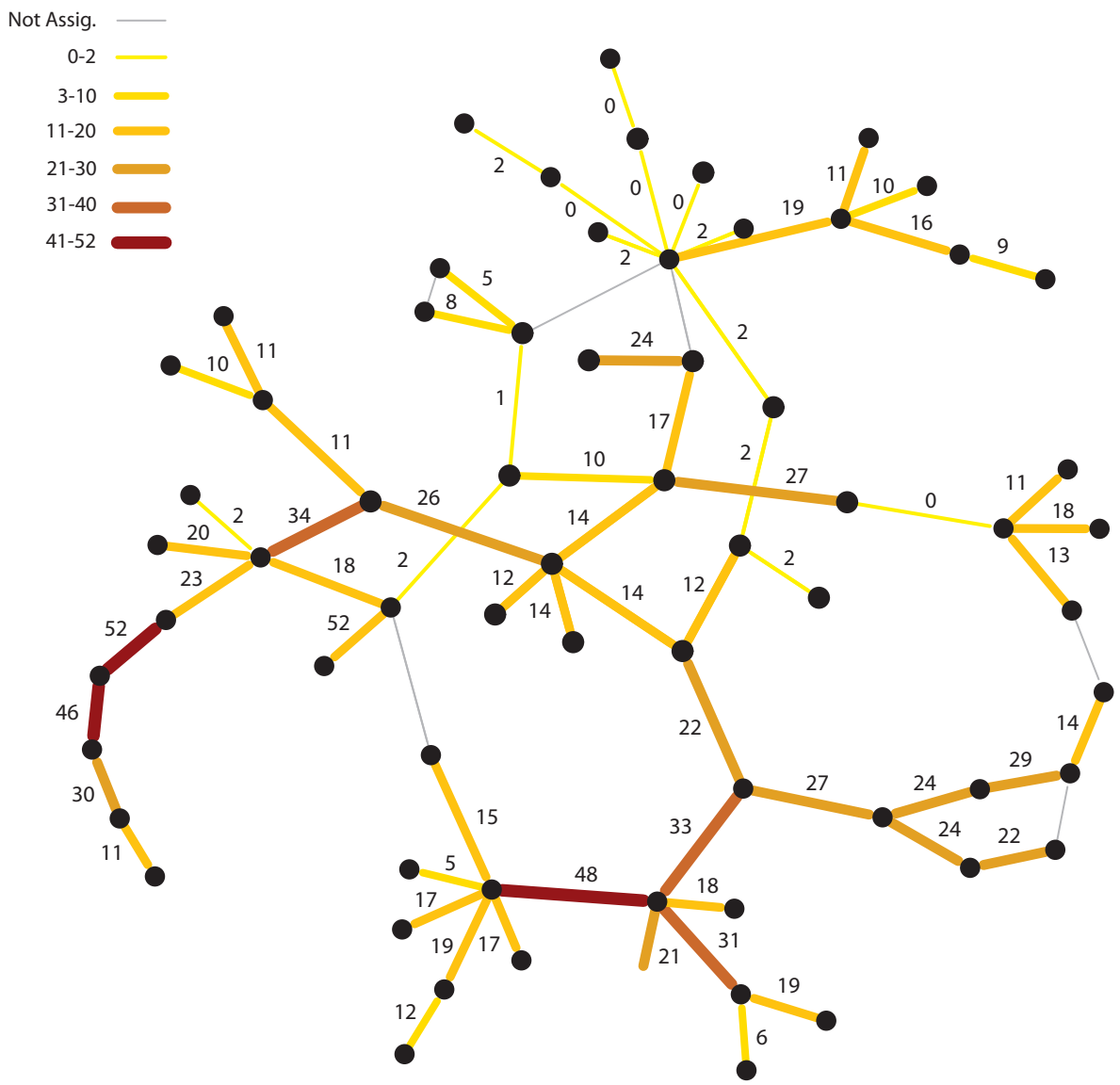

Figure 4.6: *Number of emulated link quality changes per hour. Parametrisation in Table 3.2.*

4.2 Dynamic Routing Protocols overhead in static scenarios

In order to familiarise with the different protocols and to understand protocol specific overhead characteristics a set of tentative measurements have been executed and are discussed before continuing with the presentation of advanced results illustrating the dependencies to number of nodes and dynamic link-quality changes.

The first measurement as illustrated in Figure 4.7 shows the protocol overhead of all three DRPs in a static mesh during the startup period. The protocol instances of all nodes were started at the same time. It can be seen that all protocols introduce a peak load at the beginning before stabilizing to a rather constant and continues load. The peak loads in the beginning can be explained respectively for each protocol.

- Optimized Link State Routing Protocol (OLSR) sends periodic routing updates in the form of TC messages. However, the OLSR implementation used here has the fish-eye extension activated by default. But the activation of this mechanism is delayed by a starting protocol (also by default) to speed up the convergence of the protocol when one (or several) new nodes are joining the network. As a consequence, one can observe an increased overhead over the first 150 seconds.

- The Babel protocol uses reactive routing updates that are triggered only if significant link changes are detected. Since this emulation is based on non-changing link qualities, the only significant changes are when new nodes are joining the network. Obviously this is the case when all nodes are booted. After a stabilization period of approximately 30 seconds, no further link changes are detected and the following continuous and much lower overhead can be accounted to the occasionally sent periodic routing updatates.

- The BatMan-eXperimental version 6 (BMX6) protocol is (like OLSR) emitting periodic but highly compressed routing updates at an interval of 5 seconds. However, as described in Section 2.2.3, this requires the a priori exchange of node description messages which are propagated whenever the configuration of a node changes (like after booting of protocol). The increased overhead for BMX6 in the first 20 seconds of the measurement can be accounted to this. Afterwards,

the overhead stabilizes to a very smooth average overhead of approximately 20 KBytes per second.

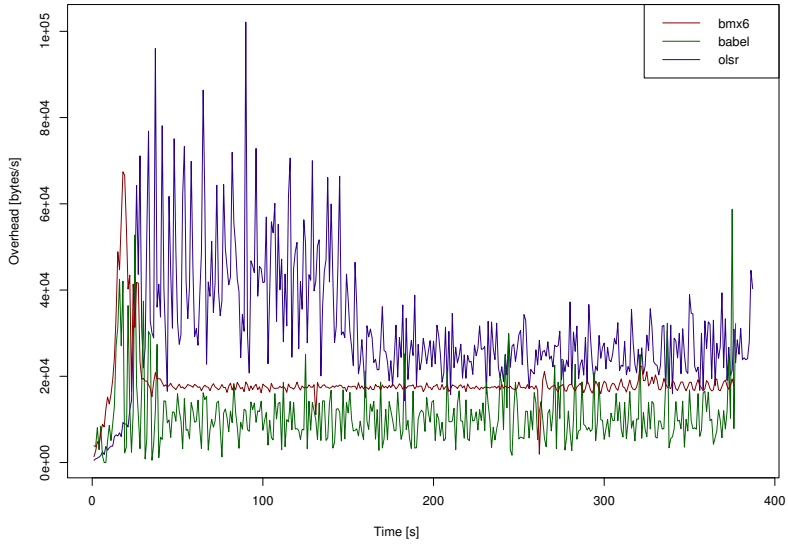

Figure 4.7: *Protocol overhead of all daemons started in parallel. Integration period: 1s. Parametrisation in Table 3.3.*

In fact, the scenario of having all nodes in a real-life Community Network (CN) being started at the same time is extremely unlikely (unless the whole CN zone is exposed to a power outage) and therefore not representative. Therefore, the temporary increased overhead resulting from the booting of DRPs will be ignored in the following. In addition to this, it must be ensured that for the convergence time measurements (discussed in the next Section), update periodicities are not in sync but equally distributed over time. To achieve this, the booting sequence of protocols has been randomized for all further measurements. An example of a randomly delayed booting sequence of **DPR!** (**DPR!**) is illustrated in Figure 4.8.

It can be seen that, due to the extended booting sequence (approximately 300 seconds), the peak values decrease. In the long term, the continues overhead of all DRPs stabilizes at the same level as measured in the previous scenario.

In the following graphs, the overhead is illustrated depending on the total number of nodes in a network. To obtain measurements for smaller networks, nodes at the edge of the topology have been removed from the original graph and only the traffic overhead

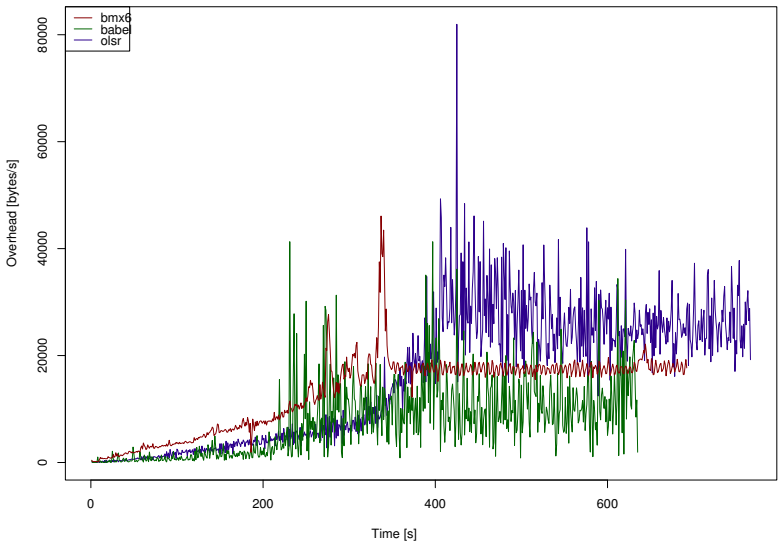

Figure 4.8: *Protocol overhead of all daemons started in serial. Integration period: 1s. Parametrisation in Table 3.4.*

following the stabilization period is considered. Figure 4.9 illustrates the total overhead depending on number of nodes. In order to understand the amount of traffic emitted by a single node or to estimate the load requirements per link, these absolute values should be divided by the number of nodes actively participating in the network. Therefore Figure 4.10 shows the same measurement results after normalization.

From the above discussed graphs it can be observed that the Babel protocol has clearly the least protocol traffic overhead. The OLSR protocol has a smaller protocol overhead compared to BMX6 in small networks (upto 40 nodes) but results in a significantly higher overhead for larger networks.

Looking carefully, one may also note that the Babel protocol has a slightly higher gradient than the BMX6 protocol. To back this presumption an extrapolation of BMX6 and Babel overhead measurement was done as illustrated in Figure 4.11. From this extrapolation it can be seen that BMX6 has the potential to outperform Babel in terms of scalability (overhead) in networks with more than 170 nodes.

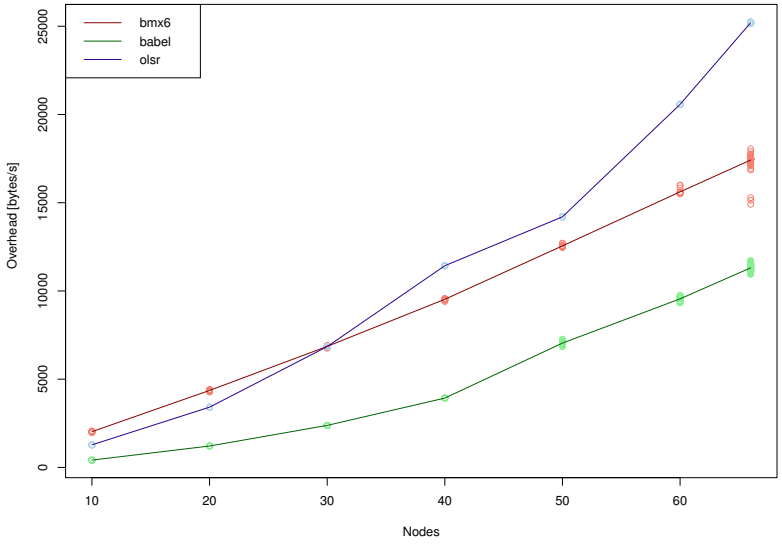

Figure 4.9: *Protocol overhead of all daemons depending on number of nodes. Parametrisation in Table 3.5.*

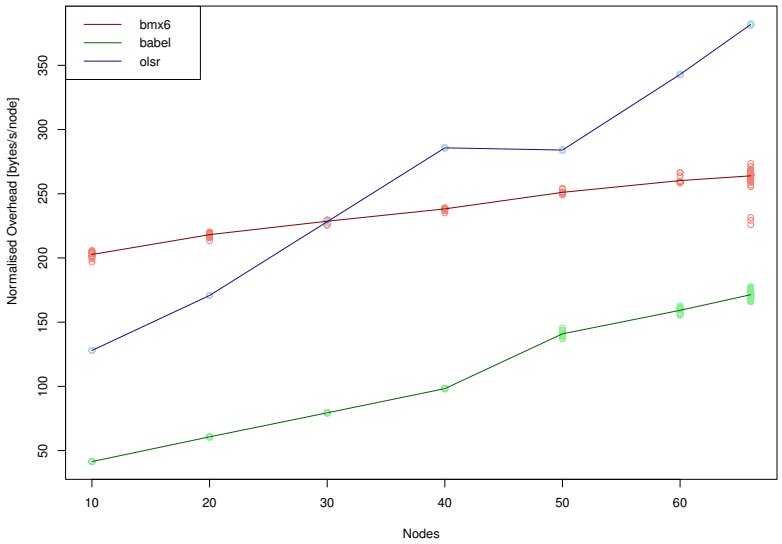

Figure 4.10: *Normalised protocol overhead of all daemons depending on number of nodes. Parametrisation in Table 3.5.*

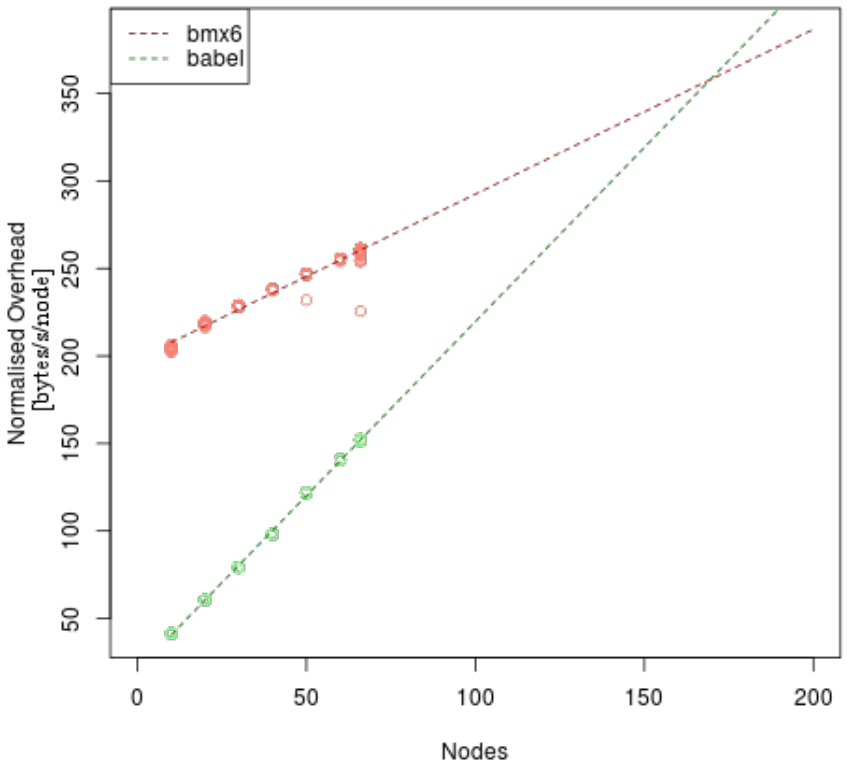

Figure 4.11: *Extrapolated normalized protocol overhead of BMX6 and Babel daemons depending on number of nodes.*

4.3 Dynamic Routing Protocols overhead in dynamic scenarios

In the following, the consequences for protocol traffic overhead due to dynamic link changes in the network are discussed.

In an effort trying to visualise the correlation between the number of concurrent link changes on the overhead of each routing protocol, both overhead and number of changes over time is illustrated for each protocol in the Figure 4.12, 4.13, and 4.14. The graphs show per protocol the overhead measured over time as well as the number of stimulated link changes at discrete moments.

Given the results of these graphs, it must be concluded that the extend of link changes as obtained from out reference network (the Barcelona guifi.net zone) has no significant influence on the overhead.

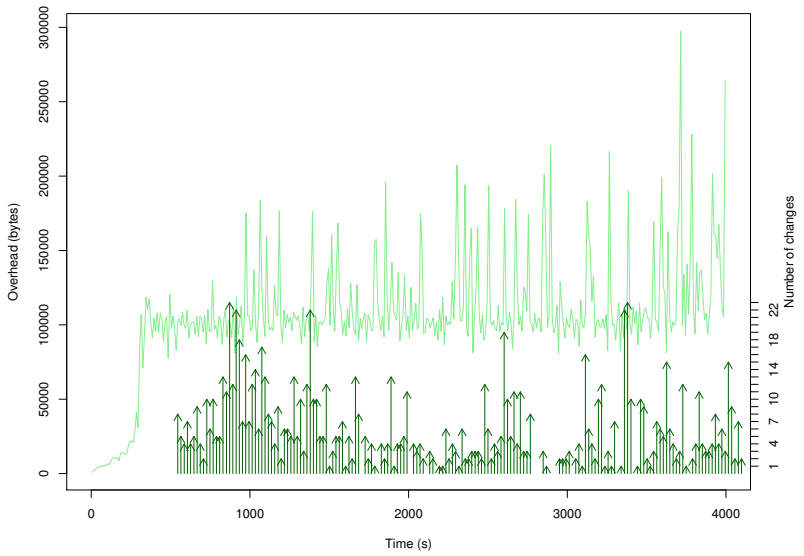

Figure 4.12: *Babel protocol overhead over time affected by discrete link changes. Parametrisation in Table 3.6.*

This conclusion is also validated by Figure 4.15 which shows the measured overhead for the static and the dynamic case in the same graph. The two lines for each protocol are so similar that they could hardly be distinguished.

For future work it would be interesting to further evaluate if overhead and link dynam-

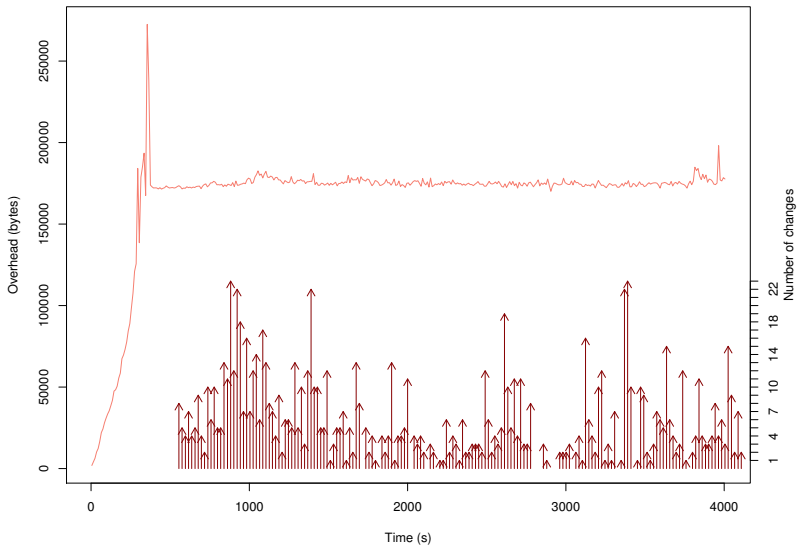

Figure 4.13: *BMX6 protocol overhead over time affected by discrete link changes. Parametrisation in Table 3.6.*

ics remains independent also for more dramatic link changes, including the complete disappearance and re-appearance of links.

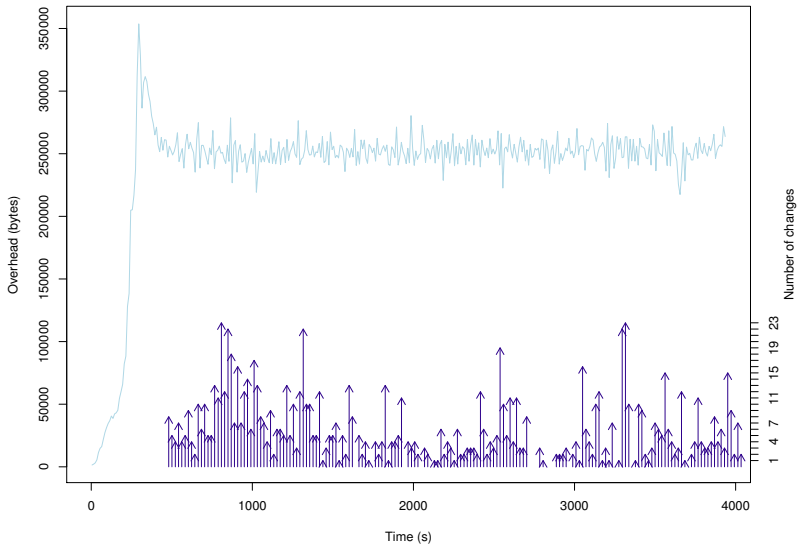

Figure 4.14: *OLSR protocol overhead over time affected by discrete link changes. Parametrisation in Table 3.6.*

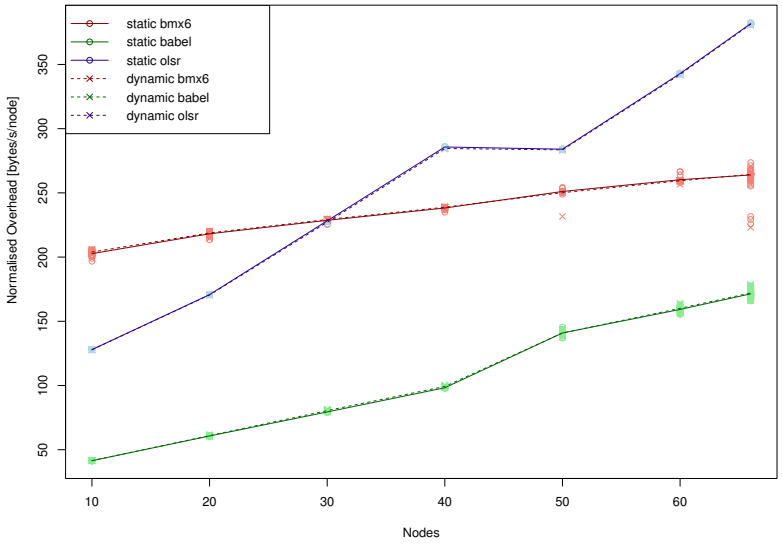

Figure 4.15: *Normalized protocol overhead of all daemons depending on number of nodes in static and dynamic scenarios. Parametrisation in Table 3.5.*

4.4 Dynamic Routing Protocols convergence time

This Section discusses the results obtained from the convergence measurements as described in Section 3.2.3. As can be seen in Figure 4.16, the convergence time of Babel and BMX6 is independent from the number of hops.

Contrary to OLSR where convergence time increases significantly for paths with eight and more hops. This can be explained with the fish-eye extension described in Section 2.2.2. Due to the sequence of TTL values used for TC messages, only every second TC message is propagated beyond the two-hop neighbourhood and only every 4 second it is propagated further than 8 hops.

In general, our measurements show that OLSR converges significantly slower than Babel and BMX6 and that BMX6 converges even faster than Babel.

The faster convergence of Babel compared to OLSR can be explains by the reactive nature of Babel, sending routing updates on demand as soon as a significant link change is detected.

The even faster convergence of BMX6 compared to Babel probably stems from the time the protocols need to locally identify link changes (in this case appearance and disappearance of a neighbouring node). All protocols use Hello messages to detect link qualities to neighbouring nodes. However, BMX6 is by default sends such messages with an interval of 0.5 seconds while Babel does this with an interval of 4 seconds. Despite BMX6 is propagating routing updates (originator messages) periodically every 5 seconds (and not reactively like Babel) these updates are not restricted by a TTL limit.

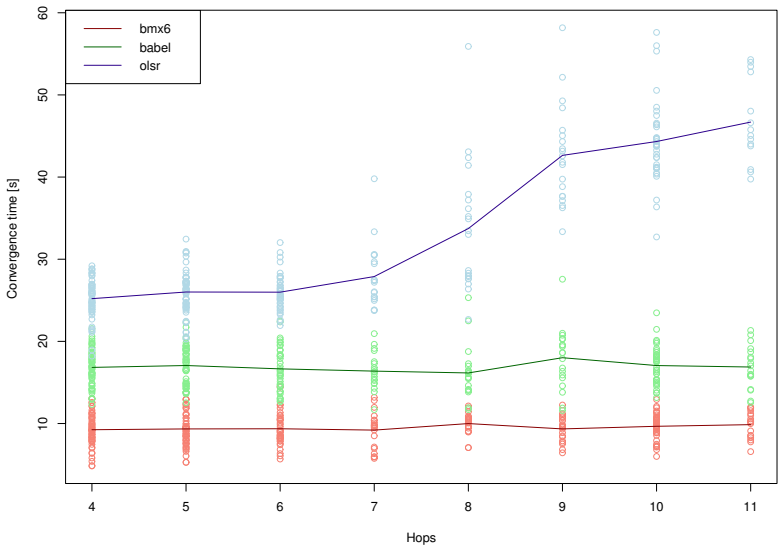

Figure 4.16: *Convergence time depending on number of intermediate hops. Parametrisation in Table 3.5 and Table 3.6.*

Chapter 5

Conclusions, future work

5.1 Conclusions

This dissertation presents the characterisation of the Barcelona zone of the guifi.net Community Network (CN) and the study of the performance of the Babel, BatMan-eXperimental version 6 (BMX6) and Optimized Link State Routing Protocol (OLSR) routing protocols for Internet Protocol version 6 (IPv6) (in terms of protocol overhead and convergence time) using an emulation environment fed with the data resulting of the characterisation.

In order to achieve these results, a number of intermediate tasks have been performed, each of them showing methodologies and revealing valuable information for the understanding and design of CNs.

In a first step, the Barcelona guifi.net zone has been characterized in terms of links and nodes relevant for the core routing (66 nodes, 69 links). To this end, static information from the guifi.net database has been combined with measurements obtained via probes send over the real (reference) network. As a result from this measurement campaigns, a comprehensive set of new and previously unknown data was collected, including continuous (one week long) traces revealing the dynamics of almost all links in terms of availability, packet loss, and delay.

This data has been post processed (and reworked) with the objective to (i) visualize it in

order to get a better understanding of the relevant parameters and (ii) make it applicable to the network emulation environment selected for the further performance evaluation of state-of-the art Dynamic Routing Protocol (DRP) implementations on top of it. Indeed, the visualisation of these data has revealed surprising findings: (i) a relation 1 to 2.5 between core-nodes and non-core-nodes (25 core nodes, 28 leaf nodes and 13 daisy-chained nodes) (ii) links perform exceptionally good taking into account that they are wireless (iii) a high correlation between guifi.net database description and the real network.

The results of the performance evaluation show that Babel introduces significantly less protocol traffic overhead compared to BMX6 and OLSR and given the characterized reference network (Barcelona guifi.net zone). However, an extrapolation of our measurement results also indicates that BMX6 has the potential to outperform Babel in larger networks with more than 180 nodes. In terms of convergence time, BMX6 proved to converge almost twice as fast as Babel and more than three times faster than OLSR. In this respect, it could also be shown that the network size has no significant influence on the convergence time of Babel and BMX6 (which is not the case for OLSR).

The emulation has also shown that the extend of link dynamics as observed during the measurement campaigns on the real (reference) network, has no significant influence on the performance of the DRPs.

As a final conclusion, the employment of the relatively young mesh routing protocols Babel or BMX6 for upcoming CN projects seem a more than promising alternative to the currently de-facto and long-term matured OLSR implementation. Given the results of our evaluation, the BMX6 protocol outperformed Babel and OLSR in terms of convergence performance and indicates a better scalability in terms of overhead for large networks with hundreds of nodes.

5.2 Future work

While analysing the results of our work we found out some interesting lines that are worthwhile further research.

Regarding the network characterisation, it would be interesting to increase the number

of measurement points in order to cross validate results. Also probing links from both edges would allow to gather information about multicast and asymmetry characteristics. Finally, applying the methodology to characterise other networks should raise information to identify best practices in CN.

Regarding DRP, performance characterising the effect of Host/Network Announcement would complement our study.

Least but not the last, it would be definitely interesting to take advantage of the opportunities that the CONFINE project must offer soon, e.g. testbeds embedded in several CNs, to run real-life experiments and contrast them with our results.

Appendix A

Data sets

A.1 Community Network characterisation

Subset 1	Subset 2	Subset 3	Subset 4	Subset 5
5076	18047	24843	46511	44148
27221	28151	17919	21209	35934
37892	17924	45223	19935	45695
26756	43363	15171	19997	40167
32281	9838	21488	18515	11308
19673	17351	8341	8451	9423
36415	35378	30132	6149	29243
36118	48024	38970	40538	16668
41757	46680	40197	38971	46725
42629	43701	39216	20262	28121

Table A.1: *Node subsets of the dynamic characterisation.*

A.2 Emulation framework

Measured link packet loss range [%]	**MLC**				
		Broadcast		**Unicast**	
	Link class	**Link delay [ms]**	**Link packet loss [%]**	**Link delay [ms]**	**Link packet [%]**
[0, 1]	3	0.1	0		
	4			0.1	0
(1, 3]	5	0.2	2		
	6			0.4	0
(3, 8]	7	0.3	5		
	8			1.6	0
(8, 15]	9	0.4	10		
	10			6.4	0
(15, 30]	11	0.5	20		
	12			25	0
(30, 60]	13	0.6	40		
	14			200	0
(60, 100]	15	0.7	80		
	16			400	0

Table A.2: *MLC link qualities discrete assignments.*

Appendix B

Interior Gateway Protocols

A network can be organized as a set of Autonomous Systems (ASs) and an Exterior Gateway Protocol (EGP) determining the reachability between ASs. An Interior Gateway Protocol (IGP) is a routing protocol used to exchange routing information within an AS. In the Internet the EGP is the Border Gateway Protocol (BGP) and the most common IGPs are Open Shortest Path First (OSPF), Intermediate System To Intermediate System (IS-IS), Routing Information Protocol (RIP) Enhanced Interior Gateway Routing Protocol (EIGRP).

B.1 Reactive and proactive DRPs

Most of the existing IGPs can be categorised into reactive protocols and proactive protocols. Reactive, also known as on-demand protocols, find a route on demand by flooding the network with *Route Requests* packets; Ad-Hoc On Demand Distance Vector (AODV) is the most well known routing protocol of this type. Proactive protocols, also known as table-driven protocols, maintain fresh lists of destinations and their routes by periodically distributing routing tables throughout the network; , OSPF and IS-IS are examples of this type of protocols. Some proactive protocols also uses *Route Requests* and some times are referred as hybrid routing protocols; RIP and Babel are examples of this case.

B.2 Link-state and distance-vector DRPs

IGPs can also be broadly categorised into link-state and distance-vector based on whether the protocol selects the best routing path by first calculating the state of each link in a path and then finding the lowest total metric path to reach a destination, or selects the best routing path according to a metric, the *distance*, and a direction, the *vector*, towards the destination.

In link-state routing protocols, each node has its own information about the complete network topology, being the information used to construct the connectivity maps the only information passed between nodes. Each node independently calculates the best next-hop to every possible destination in the network based on its local topology information and populates its routing table with the collection of best next-hops selected. Therefore, any desynchronisation of the topology view between nodes may result in routing loops, whatever algorithm and metric is used to establish the best next-hop to a destination. To prove the loop-freeness of a link-state routing protocol must be proved either that the algorithm is tolerant to desynchronized topology information or that it is loop-free when the topology information is synchronised and that this information is indeed synchronised between all nodes. Up to date any of the two prior strategies has been successfully applied on any existing link-state routing protocol and thus any routing protocol of this type has been proved to be loop-free yet. Nonetheless in practice they are widely used both in wired networks, being OSPF and IS-IS the most used IGPs by far in the Internet, and in mesh wireless networks, being Optimized Link State Routing Protocol (OLSR) the predominant in this case. Most of the link-state routing protocols, the aforementioned included, use the Dijkstra's algorithm[31] as the algorithm to find the best next-hop. In Dijkstra's algorithm each node calculates the complete shortest path to each destination[1].

In the case of distance-vector routing protocols, each nodes exchange its view of the network between its directly connected neighbours and build its view taking into account the views of the neighbours that have been previously advertised. This way, a node knows from which neighbour a route has been learnt, but it does not know where that neighbour learned the route from because any node cannot see beyond its

[1]Dijkstra's algorithm has a computational complexity of $O(n^2)$ for dense graphs, the worst case, and of $O(m + n \log n)$ for sparse graphs, being n the number of network nodes and m the number of edges.

own neighbours. Interior Gateway Routing Protocol (IGRP) and RIP[2] are examples of protocols of this type. The most common algorithm in distance-vector routing protocols is the Distributed Bellman-Forth algorithm[32]. Compared to Dijkstra's algorithm it is computationally more efficient[3], easier to implement and requires much less storage space. However Distributed Bellman-Forth algorithm is not tolerant to desynchronized topology information either; hence, modifications to force all nodes in the network to participate in some form of internodal coordination protocol are required. Destination-sequenced numbering is an efficient internodal coordination protocol used by many distance-vector routing protocols such as Destination-Sequenced Distance Vector (DSDV), AODV, Babel or any of the Better Approach to Mobile Ad-hoc Networking (B.A.T.M.A.N.) flavours. Loop-freeness of distance-vector routing protocols using destination-sequenced numbering has been proved[33].

[2]Despite v2 is considered to be and hybrid because it has properties of both types; the same occurs with EIGRP.

[3]Distributed Bellman-Forth algorithm has a computational complexity of $O(mn)$, being n the number of network nodes and m the number of edges

Acronyms

AODV Ad-Hoc On Demand Distance Vector

AP Access Point

AWMN Athens Wireless Metropolitan Network

AS Autonomous System

BGP Border Gateway Protocol

B.A.T.M.A.N. Better Approach to Mobile Ad-hoc Networking

BMXd BatMan-eXperimental daemon

BMX6 BatMan-eXperimental version 6

CN Community Network

CPE Customer Premises Equipment

CNML Community Network Mark Up Language

DRP Dynamic Routing Protocol

DSDV Destination-Sequenced Distance Vector

DSR Dynamic Source Routing protocol

EIGRP Enhanced Interior Gateway Routing Protocol

EGP Exterior Gateway Protocol

ETX Expected Transmission Count

HNA Host/Network Announcement

IID Individual IDentifier *(BMX6 specific)*

IGP Interior Gateway Protocol

IGRP Interior Gateway Routing Protocol

IPv4 Internet Protocol version 4

IPv6 Internet Protocol version 6

IS-IS Intermediate System To Intermediate System

LXC LinuX Containers

MAC Media Acces Control

MANET Mobile Ad-hoc NETwork

MLC Mesh Linux Containers

MPR MultiPoint Relays

MTR My Trace Route

OGM OriGinator Message *(B.A.T.M.A.N. and successors specific)*

OLSR Optimized Link State Routing Protocol

OLSRd Optimized Link State Routing Protocol daemon

OSPF Open Shortest Path First

RFC Request for Comments

RIP Routing Information Protocol

RTT Round Trip Time

TC Topology Control *(OLSR specific)*

TCP Transmission Control Protocol

TTL Time To Live

UDP User Data Protocol

References

[1] Bob Lantz, Brandon Heller, and Nick McKeown. A network on a laptop: Rapid prototyping for software-defined networks. *9th ACM Workshop on Hot Topics in Networks*, Nov 2010. `http://yuba.stanford.edu/foswiki/bin/view/OpenFlow/Mininet`.

[2] Axel Neumann. Investigating Routing-Protocol Characteristics with Mesh Linux Containers (MLC). Workshop, UPC, Barcelona, Spain, November 2011. `https://raw.github.com/axn/mlc/master/MeshLinuxContainers-x07.pdf`.

[3] Cloonix: dynamical topology virtual networks. `http://clownix.net`.

[4] R. Baig, Y. Bonna. *Derechos Humanos, Nuevas Realidades*, chapter El derecho a un canal de comunicación simétrico de acceso y alcance universales. Fundació Universitat Oberta de Catalunya, 2009. ISBN: 978-84-9788-805-9.

[5] Fotios A. Elianos, Georgia Plakia, Pantelis A. Frangoudis, and George C. Polyzos. Structure and evolution of a large-scale wireless community network. In *Proceedings of the 13th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2012)*. IEEE, June 2009.

[6] M. Oliver, J. Zuidweg, and M. Batikas. Wireless commons against the digital divide. In *IEEE International Symposium on Technology and Society ISTAS2010*, Australia, 06/2010 2010. IEEE, IEEE.

[7] Maria Bina. *Wireless Community Networks: A Case of Modern Collective Action*. PhD thesis, Athens University of Economics and Business, June 2007.

[8] Yann Bona. *Citizen Management of Technology: A Science and Technology Studies approach to wireless networks and urban governance trough guifi.net*. PhD thesis, Universitat Autònoma de Barcelona, December 2010.

[9] Pantelis A. Frangoudis, George C. Polyzos, and Vasileios P. Kemerlis. Wireless community networks: an alternative approach for nomadic broadband network access. *IEEE Communications Magazine*, 49(5):206–213, 2011.

[10] Filipe Dias, João Paulo Barraca, Diogo Gomes, and Rui L Aguiar. Characterization of Unplanned Metropolitan Wireless Networks. In *10th Conferência sobre Redes de Computadores*, Braga, Portugal, 2010.

[11] David Johnson, Ntsibane Ntlatlapa, and Corinna Aichele. A simple pragmatic approach to mesh routing using BATMAN. In *2nd IFIP International Symposium on Wireless Communications and Information Technology in Developing Countries*, 2008.

[12] Anna Zakrzewska, Leszek Koszalka, and Iwona Pozniak-Koszalka. Performance study of routing protocols for wireless mesh networks. In *Proceedings of the 2008 19th International Conference on Systems Engineering*, ICSENG '08, pages 331–336, Washington, DC, USA, 2008. IEEE Computer Society.

[13] Usman Ashraf, Guy Juanole, and Slim Abdellatif. Evaluating routing protocols for the wireless mesh backbone. In *Proceedings of the Third IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, WIMOB '07, pages 40–48, Washington, DC, USA, 2007. IEEE Computer Society.

[14] M. Abolhasan, B. Hagelstein, and J. C.-P. Wang. Real-world performance of current proactive multi-hop mesh protocols. In *Proceedings of the 15th Asia-Pacific conference on Communications*, APCC'09, pages 42–45, Piscataway, NJ, USA, 2009. IEEE Press.

[15] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*, MobiCom '98, pages 85–97, New York, NY, USA, 1998. ACM.

[16] I. F. Akyildiz and Xudong Wang. A survey on wireless mesh networks. *Comm. Mag.*, 43(9):S23–S30, September 2005.

[17] Eiman Alotaibi and Biswanath Mukherjee. Survey paper: A survey on routing

algorithms for wireless ad-hoc and mesh networks. *Comput. Netw.*, 56(2):940–965, February 2012.

[18] Juliusz Chroboczek. The Babel Routing Protocol. RFC 6126 (Experimental), 2011.

[19] Clausen, T., Dearlove, C., Dean, J., and C. Adjih. Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format. RFC 5444 (Experimental), 2009.

[20] T. Clausen, P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626 (Experimental), 2003.

[21] Cédric Adjih, Emmanuel Baccelli, Thomas Heide Clausen, Philippe Jacquet, and Georgios Rodolakis. Fish Eye OLSR Scaling Properties. *Journal of Communications and Networks*, 6(4):352–361, December 2004.

[22] Dang Nguyen and Pascale Minet. Scalability of the olsr protocol with the fish eye extension. In *Proceedings of the Sixth International Conference on Networking*, ICN '07, pages 88–94, Washington, DC, USA, 2007. IEEE Computer Society.

[23] P. Kuppusamy, K. Thirunavukkarasu, and B. Kalaavathi. A study and comparison of olsr, aodv and tora routing protocols in ad hoc networks. In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, volume 5, pages 143–147, April 2011.

[24] Tamilarasan-Santhamurthy. A Quantitative Study and Comparison of AODV, OLSR and. TORA Routing Protocols in MANET. *International Journal of Computer Science Issues*, 9(1):364–369, January 2012.

[25] A. Neumann and C. Aichele and M. Lindner and S. Wunderlich. Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.). Internet draft, work in progress, March 2008.

[26] Jianhua Che, Congcong Shi, Yong Yu, and Weimin Lin. A synthetical performance evaluation of openvz, xen and kvm. In *Proceedings of the 2010 IEEE Asia-Pacific Services Computing Conference*, APSCC '10, pages 587–594, Washington, DC, USA, 2010. IEEE Computer Society.

[27] *Virtualization of Linux based computers: the Linux-VServer project*, May 2005.

[28] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, pages 19:1–19:6, New York, NY, USA, 2010. ACM.

[29] ? Ieee standard for information technology-telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications. *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pages C1–1184, 12 2007.

[30] Jim Gettys. Bufferbloat: Dark buffers in the internet. *IEEE Internet Computing*, 15:96, 95, 2011.

[31] Edsger. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 1959.

[32] Bellman, R. On a routing problem. *Quarterly of Applied Mathematics*, 16, 1958.

[33] Perkins Charles E., Bhagwat Pravin. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. *SIGCOMM 94-8/94 Proceedings of the conference on Communications architectures, protocols and applications, London, England UK*, pages 234–244, March 1994.