Bachelor Thesis

Benjamin Reich

Wifi-Ad-hoc Mesh Networks for mobile Systems

Fakultät Technik und Informatik
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

Benjamin Reich

Wifi-Ad-hoc Mesh Networks for mobile Systems

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Technische Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Martin Becke
Zweitgutachter: Prof. Dr. Thomas Lehmann

Eingereicht am: 28. March 2022

Benjamin Reich

Thema der Arbeit

Wifi-Ad-hoc Mesh Networks für mobile Systeme

Stichworte

WLAN, Mesh, autonome Ad-hoc-Netze, Routing Protokoll, Raspberry Pi

Kurzzusammenfassung

Diese Bachelor Thesis vergleicht verschiedene Ad Hoc Netzwerk Protkolle für mobile autonome Systeme. Die größte Herausforderung ist, dass die Netzwerk Topologie jederzeit verändert werden kann. Dienste sollten dezentral organsiert werden, da ein System jederzeit die Verbindung zum Netzwerk verlieren kann. Solch ein Netzwerk kann für koordinierte Multi-Roboter Exploration genutzt werden. Zum Teilen von Informationen über das Netzwerk werden verschiedene Wlan-Mesh-Routing Protkolle miteinander verglichen in Durchsatz und Robustheit. Verglichen werden diese Protokolle mit einer Emulation auf einen Computer und verifiziert auf Rasperberry Pis.

Benjamin Reich

Title of Thesis

Wifi-Ad-hoc Mesh Networks for mobile Systems

Keywords

Wifi, Mesh, MANET, Routing Protocol, Raspberry Pi

Abstract

This bachelor thesis compares different ad hoc network for mobile autonomous systems. The biggest difficulty is that the network topology changes at any time and changing network characteristics. Services should also be set up decentralized, as systems can lose their connection to the network. Such a network can be used for Coordinated Multi-Robot Exploration. For sharing information across the network. The network should

be as large as possible, contains 4 nodes and allow data transfer speed of ∼100Mbit/s. Different existing WiFi-Mesh-Routing-Protocols will be compared in throughput and resilience. The comparison is emulated on a computer and verified on Rasperberry Pis.

Contents

List of Figures

List of Tables

Acronyms

ARM Advanced RISC Machines.

CaDS Communication and Distributed Systems.

DHT Distributed Hash Table.

DNF did not finish.

IP Internet Protocol.

LAN Local Area Network.

Loomo Segway Loomo Robot.

LoRaWAN Long Range Wide Area Network.

MANET mobile ad hoc network.

OLSRv2 Optimized Link State Routing Version 2.

STP Spanning Tree Protocol.

tc traffic control.

TCP Transmission Control Protocol.

VPN Virtual Private Network.

WLAN Wireless Local Area Network.

1 Introduction

Computer communication and distributed computing are central to the development of the modern world. Collecting, sharing and processing information between different nodes in a distributed computing system are fundamental tasks. Such distributed computing systems are built on top of computer networks. [1]

Computers can be found in almost every device these days. These computers can be connected to each other in a computer network. This brings different requirements, depending on the scenario, for different computer networks. Many transmission media and protocols have been developed to adapt to different use cases, which impose constraints on transmission speed, latency, range, resilience etc.

Possible use cases for a network include for sharing information across the network and coordination in autonomous systems like in car-to-car communication [2] or in communication in disaster areas [3].

A mobile ad hoc network (MANET) is a special type of computer network with idiosyncrasies for the use in mobile wireless networks. [4] The Communication and Distributed Systems (CaDS) group[1] uses such a MANET in an autonomous multi-robot exploration system [5] built on top of a Segway Loomo Robots (Loomos). In this scenario, the MANET enables inter-robot communication in environments, which impose a random, rapidly changing, multi hop topology [4]. In this work different MANET routing protocols are compared to identify the most suitable option for the given scenario of the CaDS group[1].

[1] https://cads.informatik.haw-hamburg.de, accessed September 2021

1.1 Motivation

The CaDS group uses a MANET[4] to route messages between nodes for multi robot exploration[5]. Nodes are Loomo, which share their information with other nodes. In the exploration process the network topology will often change to structural conditions.

Critical to the operation of mesh networks is the routing of messages between nodes, for which efficient paths between source and target nodes need to be found. A MANET [4] has the additional difficulty that topology could change quite often. The routing protocol must be able to handle this. This means every routing information has to be updated when a new topology is available. Such a mesh network could be used in car-to-car communication [2]. Cars could benefit from information of other participants. They could share information about pedestrian or road information. Another use case could be a decentralized network in emerging markets or disaster areas [3]. Every smartphone could be part as a router in the network.

Efficient and robust path finding in a graph is a problem in the field of network routing. Flooding is a common approach to solve this problem. However, its disadvantages, which are its inability to handle loops and its computational inefficiency, may be prohibiting factors in some scenarios, such as overhead in message handling. Routing protocols are a set of algorithms that allow efficient path finding in graphs. This thesis compares different routing protocols and their applicability to MANETs.

MANETs[1, 4] are a type of network that does not rely on fixed infrastructure topology. In general, mobile ad hoc networks include numerous mobile wireless communication devices that are geographically distributed across an area and are communicatively coupled to each other via a wireless communication channel. It allows faster and more frequent topology changes than a traditional network. More specifically, a mobile ad hoc network may be a multi-hop network which enables each of the mobile wireless communication devices to route data to and from the other mobile wireless communication devices without the need for infrastructure to route the data between the mobile wireless communication devices. These ad-hoc networks are applicable to a wide variety of problems, such as car-to-car communication for sharing information and coordination in autonomous systems [2]. Furthermore, MANETs may be used in communication in disaster areas [3] or in other scenarios which could benefit from a decentralized mesh network.

The CaDS group[2] uses such a MANET in an autonomous multi-robot exploration [5] system built on top of Loomos. In this scenario, the MANET enables inter-robot communication in environments, which have not been covered with fixed communication topology infrastructure. Therefore, non-decentralized networks are not very suitable, like Wireless Local Area Network (WLAN) in infrastructure mode [6].

As nodes that form a MANET are not fixed to a particular location, the availability of individual links between nodes may change rapidly over time, when nodes are moving. This poses a challenge to traditional routing algorithms, which rely on the assumption that network structures remain relatively constant. A traditional network does not normally change its topology that often in a short time [1]. Therefore, dedicated routing algorithms are required for a MANET to function efficiently. This thesis analyzes different routing algorithms and whether they can be applied to MANETs in an attempt to identify characteristics, which make algorithms suitable for these types of networks.

1.2 Goals and Objectives

The main goal of this thesis is to compare different routing protocols, which are based on different algorithms and implementations. Different metrics are used to evaluate the performance of these routing protocols in different scenarios. There are other radio protocols which are not considered in this thesis. The reason is that these are often not Internet Protocol (IP) based or do not have the necessary bandwidth(100 Mbit/s). Bluetooth, ZigBee, LoRaWAN and Thread are some of these protocols which are not considered.

All routing protocols used have to be actively developed and provide support for IPv6.

In some cases, characterizations only become visible in larger networks. Therefore, larger networks are emulated with ***Mesh Network Lab***[3] from *Moritz Warning* and *contributors*. Later, the results will be validated with a setup on Raspberry Pis. The Raspberry Pi is common used single board computer, with big a community of users. This can also be used with the Loomo, since it gets power and Ethernet (USB-OTG) via the USB interface.

[2] https://cads.informatik.haw-hamburg.de, accessed September 2021
[3] https://github.com/mwarning/meshnet-lab, accessed September 2021

1.3 Structure

Chapter 2 explains fundamental concepts of routing protocols and introduces the used MANET routing protocols. This includes static and dynamic routing protocols. Also, the theory of operation of each MANET protocol is described. In Chapter 3 is the procedure of the thesis explained. It describes the different experiment scenarios and gives an expectation of the results in these various scenarios. Chapter 4 presents and discusses the results of the procedure. Finally, in Chapter 5 of this thesis summarizes the results and considers the results for the given use case. A further outlook of future work on this topic is provided.

2 Theory and Related Work

This chapter provides a common understanding for the remaining of the thesis. It introduces terminology that will be built upon later.

2.1 Routing Protocols

The main function of a network is to route packets from one node to another. Often several hops are required to reach the destination node from the source node. The algorithm which chooses the next hop is called a routing protocol. The following section about routing protocols is based on Computer Networks[1] from *Andrew S. Tanenbaum* and *David J. Wetherall*.

2.1.1 Static Routing

Static routing is a common and simple way to determine which way to choose for each packet. It is used to forward packets to the next hop. The next hop is selected by the router based on the destination address. The information about how to select the next hop are listed in the routing table. The disadvantage of this approach is that the routing table is static. Every time a new node joins the network, the routing table must be updated manually. [1, Chapter 5.2]

2.1.2 Flooding

The simplest way for routing is flooding [1, Chapter 5.2.3]. This is used to forward all incoming packets to all other nodes. Flooding creates generates lots of duplicated packets. Without a hop counter, the possible number of packets is infinite. This counter could be a part of the packet header, which is decremented at each hop, and the packet is discarded

when the counter reaches zero. This approach could be improved by keeping track of which packets already have been flooded, to avoid transmitting duplicated packets.

A better way of stopping floods is to have routers keep track of which packets have been flooded, so they do not send them out a second time. One way to achieve this goal is having the source router add a sequence number in each packet it receives from its hosts. Each router then needs a list per source router telling which sequence numbers originating at that source have already routed. There are however, still problems how long sequence numbers should be stored.

2.1.3 Dynamic Routing

A dynamic routing protocol is more flexible than a static routing protocol. Dynamic routing protocols allow changes in the routing information, such as advertisements or routing table entries. For example, dynamic routing protocols may determine whether a router receives advertisements from other routers and make routing table entries based on this determination.

Distance Vector Routing

A Distance Vector Routing protocol [1, Chapter 5.2.4] is a more complex routing protocol than flooding. The advantages are that it is dynamic, which is more efficient, while providing the shortest path to the destination. The Bellman-Ford algorithm is commonly used to compute the shortest path in this type of protocol. Each router in a distance vector routing network keeps a routing table that is indexed by and has one entry for each router in the network. This entry is divided into two sections: the preferred outgoing line for that destination and a distance estimate to that destination. The distance can be calculated using the number of hops or another metric. The router is believed to be aware of each of its direct neighbours' distance. The distance is one hop. Other metrics could also be used, e.g. the propagation delay or bandwidth.

Link State Routing

A Link State Routing Protocol [1, Chapter 5.2.5] is also a dynamic routing protocol. It is also used to compute the shortest path to the destination. It aims to converge faster than

the distance vector routing protocol, but it is also more complex. Link State Routing Protocols are often based on the Dijkstra algorithm.

Compared to distance-vector routing, link state routing optimizes routing structures by performing calculations on the best routes instead of just passing full routing table information between routers. Link state routing only sends interface information about the different interfaces existing on a router and the networks to which the router is connected. Therefore, rather than send all available routing tables, link state updates send only the information about the learned routes.

Each router needs to follow these steps to work:

1. Discover all neighbours and learn their network information.
2. Calculate the cost for each neighbour, e.g. hops.
3. Send the new learned information to all neighbours and receive packets from all other routers.
4. Compute the shortest path to every other router.

2.1.4 MANET Routing Protocols

MANET routing protocols are a special type of Dynamic Routing. They are optimized for mesh networks, which are often wireless, and their topologies change frequently. The following subsection will introduce the protocols that are evaluated in this thesis. All of them support IPv6 and are still under development. These two requirements must be fulfilled. Each of these algorithms performs differently in certain scenarios and has been optimized for certain use cases, which may make them more or less applicable to the Loomo use case.

Babel

Babel [7, 8] is based on Distance Vector Routing and is an IETF standard[1]. It supports IPv4 and IPv6 networks and works on ISO/OSI-Layer 3. The source code is also available on GitHub[2].

[1] https://datatracker.ietf.org/group/babel/about, accessed February 2022
[2] https://github.com/jech/babeld, accessed February 2022

Batman-adv

Batman-adv [9] is the short form of B.A.T.M.A.N. advanced. It is based of the B.A.T.M.A.N. algorithm, which is itself based on Distance Vector Routing. An older implementation of B.A.T.M.A.N., also known as batmand, works on the ISO/OSI-Layer 3. The newer batman-adv works on the ISO/OSI-Layer 2, which is a part of the Linux kernel[3]. Because batman-adv operates on ISO/OSI-Layer 2, it is independent of the IP implementation. The source code is also available in a public Git repository[4].

BMX7

BMX7 [10, 11] is the successor of BMX6, which is no longer under development. BatMan-eXperimental Version 6 (BMX6) was the branch for testing new features and concepts of batmand (predecessor of batman-adv). Over the time a re-integration was not feasible anymore. The project was therefore carried out independently of B.A.T.M.A.N. [12]. BMX7 adds additional security features on top of BMX6. It supports IPv4 and IPv6 networks, works ISO/OSI-Layer 3 and is based of on a Distance Vector Routing. The source code is also available on GitHub[5]. In this work BMX7 is found as bmx7.

OLSR2

Optimized Link State Routing Version 2 (OLSRv2) [13, 14] is based on Link State Routing. In this thesis the name olsr2 is used. It supports IPv4 and IPv6 networks and works ISO/OSI-Layer 3. The source code is also available on GitHub[6].

Cjdns

Cjdns [15] is a routing protocol is based on Distributed Hash Tables (DHTs). It works as overlay network, comparable to a Virtual Private Network (VPN) or the Tor Network.

It supports end-to-end encryption. The source code is also available on GitHub[7].

[3] https://www.kernel.org/doc/html/latest/networking/batman-adv.html, accessed February 2022
[4] https://git.open-mesh.org/batman-adv.git, accessed February 2022
[5] https://github.com/bmx-routing/bmx7, accessed February 2022
[6] https://github.com/OLSR/OONF, accessed February 2022
[7] https://github.com/cjdelisle/cjdns, accessed February 2022

Yggdrasil

Yggdrasil [16] is a routing protocol that has been introduced in 2018. It is heavily based on DHTs and Spanning Tree Protocol (STP), which is often applied to Local Area Networks (LANs) on ISO/OSI-Layer 2 [17]. It is heavily inspired by Cjdns. So it also acts as an overlay network. It is encrypted end-to-end as well. The source code is also available on GitHub[8].

2.2 Linux Network Namespaces

Linux Network Namespaces[9] are helpful to emulate bigger networks without the need to have multiple or hundreds physical nodes in network. It reduces the upfront cost required for evaluation of protocols.

Linux Network Namespaces are one of many varieties of namespaces in the Linux operating system. Namespaces in general are used to offer environments to isolate processes. Network namespaces are logically a copy of the network stack. They provide their own network devices, routing tables, and other network-related information.

[8] https://github.com/yggdrasil-network, accessed February 2022
[9] https://man7.org/linux/man-pages/man7/namespaces.7.html, accessed September 2021

3 Procedure

Different mesh network technologies are compared using a variety of metrics, such as scalability, bandwidth and convergence. As it is infeasible to compare these metrics on actual hardware, the networks are emulated. To recognize certain characteristics and behaviors of a mesh networking technology, larger network topologies are required. Due to the high number of nodes and topology changes necessary for this, it is not possible to compare this on real hardware in a reasonable amount of time. To solve this problem it is either possible to simulate or to emulate the networks. Simulations require the reimplementation of each mesh network protocol within in the simulation environment. Therefore, the emulating approach is recommended.

3.1 Network Emulation

For emulating multiple network nodes Mesh Network Lab[1] from *Moritz Warning* and *contributors* is used. It is based on the Linux Network Namespaces, which are explained in section 2.2. Traffic control (tc)[2] is used to apply different characteristics to the network, e.g. latency, bandwidth and packet loss. Experiments with different network topologies are performed to find out how different they will perform in specific scenarios. Mesh Network Lab creates these networks using user defined network specification files.

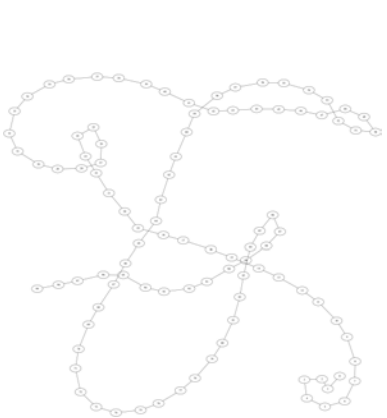
3.1.1 Experiment Topologies

In the experiments different topologies are used. These topologies have different characteristics, that pose different challenges to routing algorithms.

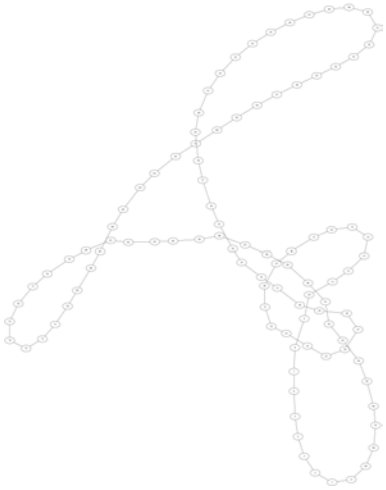
[1] https://github.com/mwarning/meshnet-lab, accessed September 2021
[2] https://man7.org/linux/man-pages/man8/tc.8.html, accessed September 2021

A line topology (Figure 3.1a) has been chosen as one experiment topology. Each node has a single predecessor and a single successor, except for the first and the last node, which only have a successor or predecessor respectively. This type of topology is expected to be the worst case scenario for a mesh network. It is suspected the MANET protocols will have difficulties to converge in this topology. In a 100 node setup, the information of the first node has to pass through 98 other nodes before it reaches the last node.

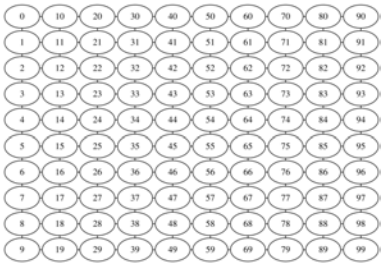
In a circle topology (Figure 3.1b), each node is connected to the next node in the circle. In this type of topology MANET protocols should have fewer difficulties than in the line topology. The longest optimal path in a circle topology with 100 nodes is 49 hops. This results in shorter expected paths compared to the line topology.



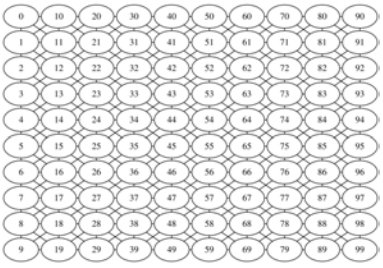
(a) Line topology with 100 nodes



(b) Circle topology with 100 nodes



(c) Grid4 topology with 100 nodes



(d) Grid8 topology with 100 nodes

Figure 3.1: Topologies with 100 nodes

In a grid4 topology (Figure 3.1c), each node is connected to multiple neighbours. Each Node has four neighbours, two vertical and two horizontal, except for the nodes on the edges.

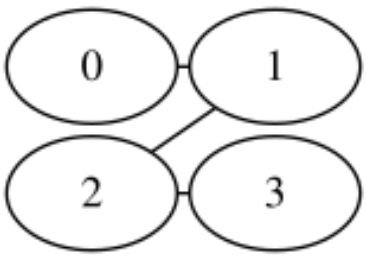
The grid8 topology (Figure 3.1d) is an extension of the grid4 topology. Each node is connected to eight neighbours. There are two vertical, two horizontal and 4 additional diagonal links. Edge nodes again do not have all of these links.

These two topologies should benefit from the distributed computing of the MANET protocols. The longest optimal path in grid4 topology with 100 nodes is 19 hops. The grid8 topology hast the longest optimal path of 9 hops.

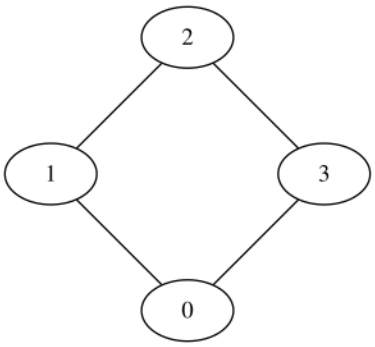
As the link count heavily increases compared to the line/circle topology, the computational cost may increase, depending on the algorithm.

Each network has 100 nodes, as shown in figure Figure 3.1.

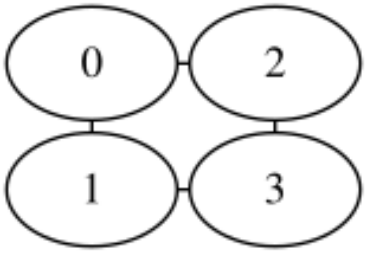
To verify that the measurements reflect real world conditions, the emulated networks are shrunk to a smaller network size of four nodes, as shown in figure 3.2. These measurements can be compared with the Validation section. The reduced node number a grid4 topology and a circle topology with 4 nodes are the same.



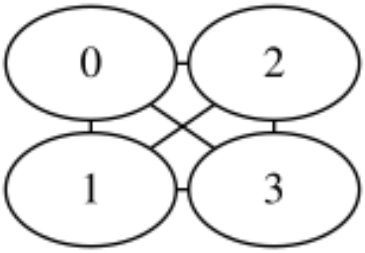
(a) Line topology with 4 nodes



(b) Circle topology with 4 nodes



(c) Grid4 topology with 4 nodes



(d) Grid8 topology with 4 nodes

Figure 3.2: Topologies with 4 nodes

3.2 Scalability Validation

Using Linux Network Namespaces heavily relies on the used hardware. Scaling the number of emulated nodes is CPU bound. To ensure that subsequent measurements are not affected by hardware bottlenecks, the following steps are taken:

1. Create a grid4 with the desired number of nodes.
2. Start the MANET service for each node.
3. Wait for 30 seconds.
4. Select as many random ping paths as contained nodes
5. Ping the paths and save the results.
6. Abort, if the less than 40 % of the nodes are not reachable.
7. Start at item 1 with a bigger grid of nodes.

As soon as the number of nodes or links becomes too large, the percentage of hosts reached is expected to decrease. This validation ensured that 100 nodes can be handled by the chosen hardware for the evaluation.

3.3 Convergence Analysis

How fast a network converges in mesh networks is a key metric and performance indicator of a routing protocol. This metric is called the convergence time of the network. MANET protocols with a faster convergence time should in principle be able to react faster to topological changes. After it detects a topological change, a faster convergence should imply all nodes can be reached sooner than with a slower one.

Different network characteristics are applied with tc to the network on each link. This should reflect the characteristics in a real world WLAN. The characteristics shown in Table 3.1 are applied.

Environment	Latency	Bandwidth	Packet Loss
Environment 1	1 ms	100 Mbit/s	-
Environment 2	10 ± 5 ms (normal distribution)	100 Mbit/s	-
Environment 3	1 ms	100 Mbit/s	random 2 %
Environment 4	10 ± 5 ms (normal distribution)	100 Mbit/s	random 2 %
Environment 5	10 ± 5 ms (normal distribution)	100 Mbit/s	random 0.2 %

Table 3.1: Network Environment Parameters

1 ms is a typical latency for a link in an optimal environment. 10 ± 5 ms are the typical latencies for a wireless link in an environment with wide distances. The normal distribution is used to represent the variation of the latency in the real world behavior. 2 % packet loss are can be found wireless link in an environment with wide distances and obstructions or interference over each link. 0.2 % is more moderate approach to the packet loss in a more favorable environment.

For the convergence analysis, the following steps are taken:

1. Create a network.
2. Start/restart the MANET service for each node.
3. Wait for 2 seconds.
4. Select as many unique random ping paths as contained nodes.
5. Ping these unique paths and save percentage of successful pings.
6. Go back to step 2 and increase the time to wait in step 3 by an additional 2 seconds until 120 seconds are reached.

These steps a repeated for the previously introduced network topologies and different network characteristics are applied with tc.

3.4 Bandwidth Analysis

If the bandwidth is affected by the routing protocol, it may be unsuitable for the Loomo scenario that this thesis evaluates for. Therefore, the bandwidth of each protocol is measured. The measurements of the bandwidth in the networks are made with iperf3[3].

Each bandwidth measurement is done in isolation between two nodes. Therefore, only one path in network is measured not the overall bandwidth. Each measurement runs for 100 seconds over a single Transmission Control Protocol (TCP) connection.

3.5 Validation

To ensure the emulation works on real world conditions, some experiments are repeated on Raspberry Pis. Raspberry Pis were chosen because they are inexpensive, and they can be powered by Loomo without an external power supply. This will also be the hardware that will later be equipped on the Loomos. Each Raspberry Pi is connected to via Ethernet for management purposes. The actual MANET is connected over WLAN with a single MANET protocol at time. This setup is shown in Figure 3.3.

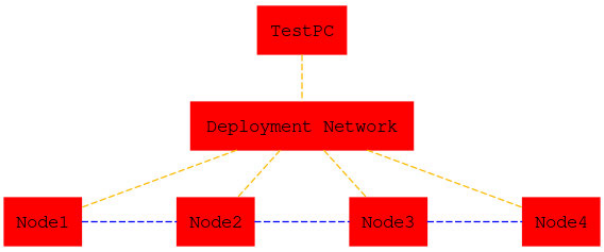


Figure 3.3: One possible example validation setup with four nodes in a line topology

3.6 Expectation

The Scalability Validation should ensure that 100 nodes can be handled on the given hardware. Otherwise, the subsequent results would not be reliable.

The expectation of the Convergence Analysis is that with each protocol, as time goes on, the percentage of successful pings will increase. This should apply to each topology and

[3] https://iperf.fr/, accessed February 2022

applied characteristics. Over time, all nodes should be able to exchange their routing information with each other. As successful pings are measured, and random paths are used to determine the convergence, it is possible that results are noisy. Furthermore, packet loss adds noise to the measurements. However, a trend should be recognizable.

The overall Bandwidth Analysis should not be affected by protocols or the used topology. Some bandwidth is needed to exchange routing information for the MANET to function, but this should not significantly impact the overall bandwidth. It is possible that encryption or the encapsulation of the data (depends on the MANET protocols) can impact the bandwidth.

The results of the Validation should be similar to an emulation with 4 nodes. It is only assumed that the number of nodes is too low to make a statement. The metrics in such small networks can be so small that they are difficult to measure in both the emulated and real world environment with the methods mentioned above.

4 Results

This chapter contains the results of the comparison of the MANETs protocols using the metrics described in chapter 3.

4.1 Hop Limit

One of the key findings was the hop limit in the used protocols. Some of these hop limits were not well documented and could only be determined by trial and error. In spite of the limitations, it is expected that cjdns will perform much worse than the other protocols in the 100 nodes setups. The pinged paths in Convergence Analysis can be longer than the hop limit. Batman-adv and bmx7 can be also affected by this. The discovered hop limits are summarizes in Table 4.1.

The IP hop limit for IPv4 and IPv6 is 2^8 [18, 19].

This work emulated all ping paths in the experiments with no hop limit.

	Hop Limit
babel	99
batman-adv	32
bmx7	32
cjdns	14
olsr2	99
yggdrasil	99

Table 4.1: Hop Limits identified in a Line Topology with 100 nodes

The hop limit of batman-adv has also been confirmed by the developers[1].

[1] https://lists.open-mesh.org/mailman3/hyperkitty/list/b.a.t.m.a.n@lists.open-mesh.org/thread/AZBZGHSV7GQ5D7I6BF7C3J3AILCBCZ5L/, accessed March 2022

4.2 Scalability Validation

The Figure 4.1 show the results of the scalability validation. The used hardware was able to handle the emulation for 100 nodes without bottlenecking. Cjdns performs the worst, this can probably be attributed to the low hop limit of 14. This protocol does not have the possibility to reach nodes that are further away than 14 hops.

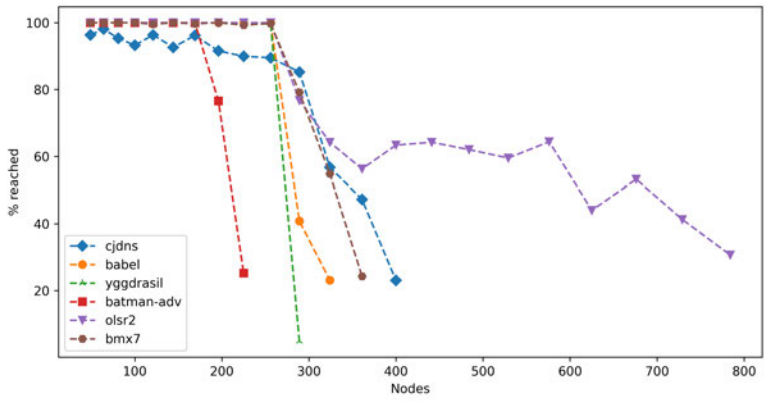


Figure 4.1: Benchmark of used protocols

4.3 Convergence Analysis

Due to the fact that all runs on the existing hardware require 2 weeks for all permutations (topologies and link parameters), all experiments for the convergence analysis were run only once. The Average values for multiple execution would very likely lead to a more accurate statement.

4.3.1 Delay with 1ms

In an optimal environment, where each link has 1 ms delay, the convergence time of the MANETs protocols are nearly the same in a grid (either grid4 (Figure 4.2) or grid8 (Figure 4.3)).

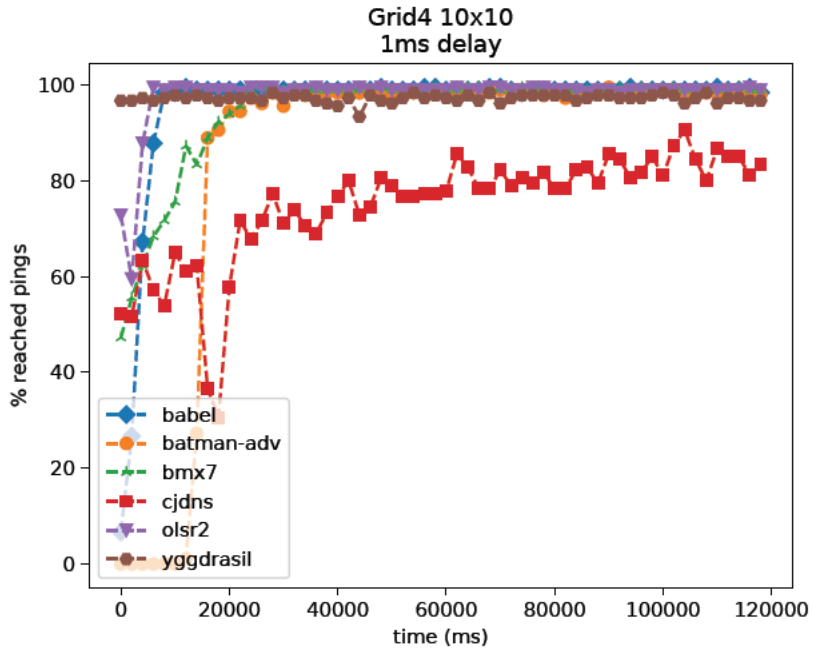


Figure 4.2: Optimal environment with 100 nodes in a grid4 topology

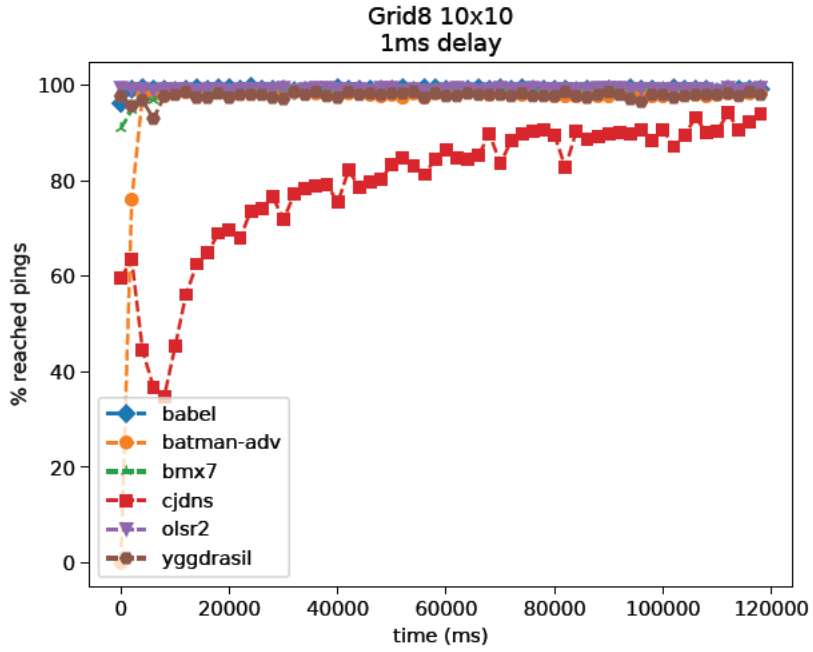


Figure 4.3: Optimal environment with 100 nodes in a grid8 topology

Babel, olsr2 and yggdrasil can handle the line (Figure 4.4) and circle ((Figure 4.5)) topology much faster than the other competitors. Babel and olsr2 also reach the highest percentage of convergence in the line and circle topology. This may be attributed to the higher hop limit.

In comparison, between Figure 4.3 and Figure 4.4, can be recognized that MANET protocols benefit from many links to each other. The convergence time for each protocol in the gird8 topology is much faster and reaches more nodes than in the line topology.

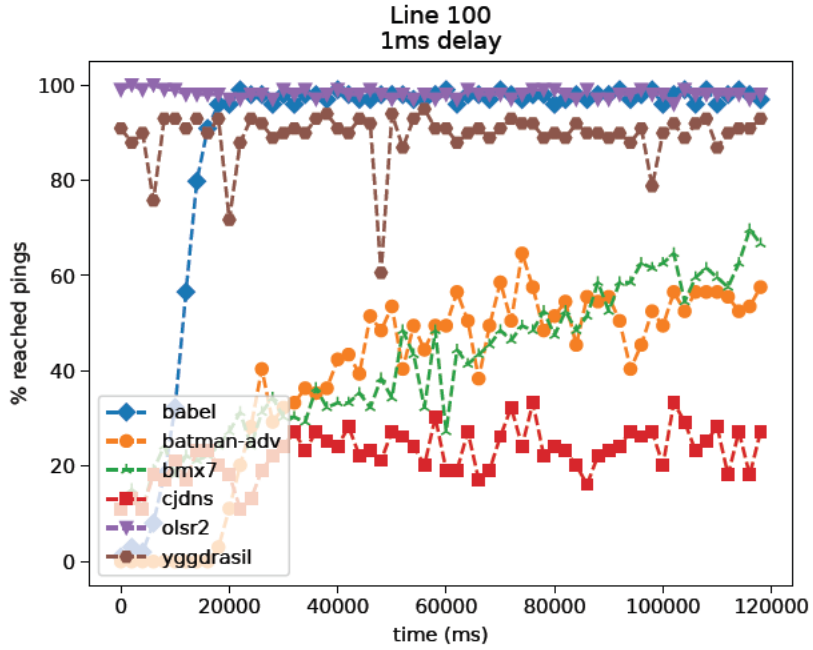


Figure 4.4: Optimal environment with 100 nodes in a line topology

As explained at the beginning, such spikes or drops as can be seen in Figure 4.4 (yggdrasil), are eliminated in several runs by calculating the average.

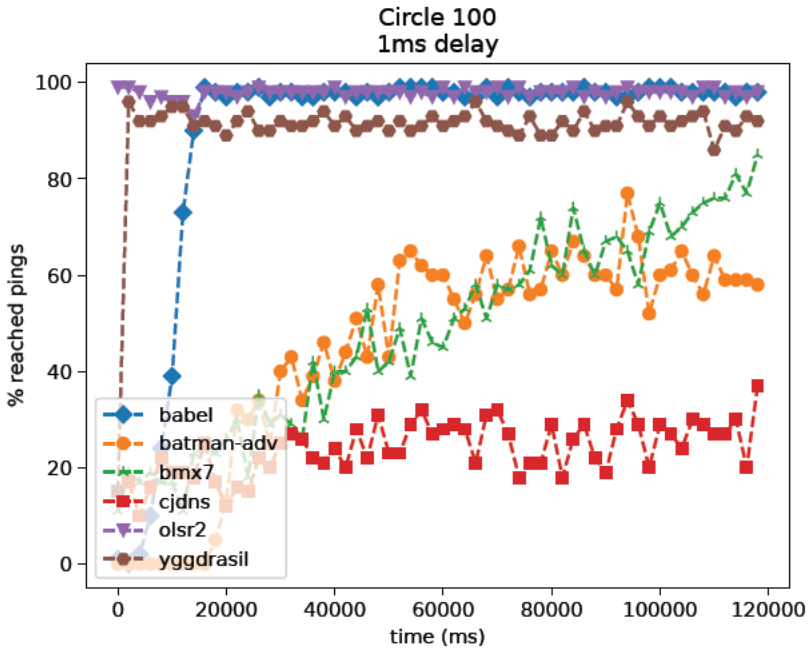


Figure 4.5: Optimal environment with 100 nodes in a circle topology

4.3.2 Delay with 10ms ± 5ms

In an environment with fluctuating latencies 10 ± 5 ms (normal distribution). The grid topologies (either grid4 (Figure 4.6) or grid8 (Figure 4.7)) perform nearly identically in an optimal environment. This does not apply for the yggdrasil protocol. This protocol is not able to reach a convergence over 80 % and underperforms cjdns, which has a much lower hop limit.

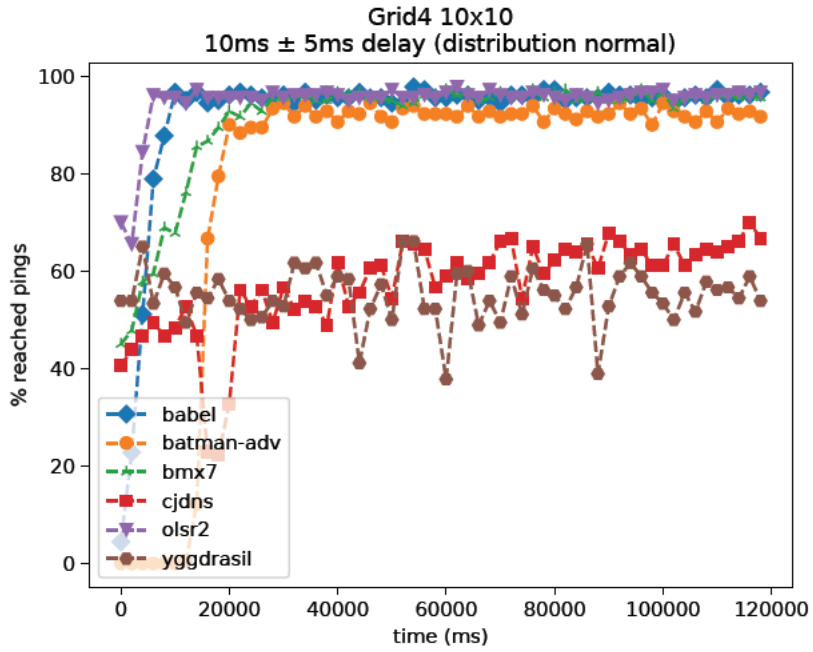


Figure 4.6: Environment with fluctuating latencies and 100 nodes in a grid4 10x10 topology

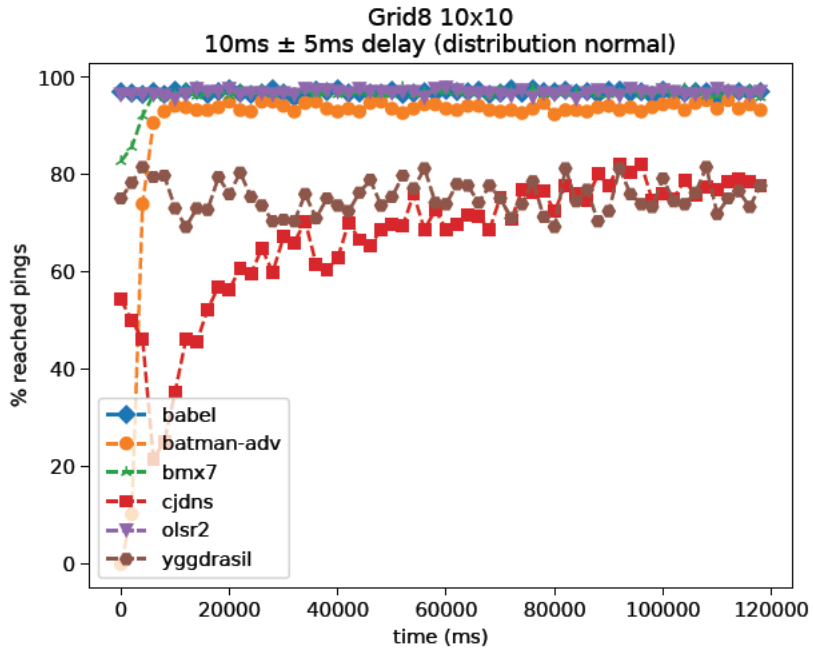


Figure 4.7: Environment with fluctuating latencies and 100 nodes in a grid8 10x10 topology

Atypical is that with olsr2 in the line (Figure 4.8) and circle (Figure 4.9) topology, the convergence falls off over time. These behaviours were reproducible in the environment used. Babel and olsr2 can reach a convergence over 80 % in the line and circle topology. These performed 20 to 60 percentage points better that the competitors.

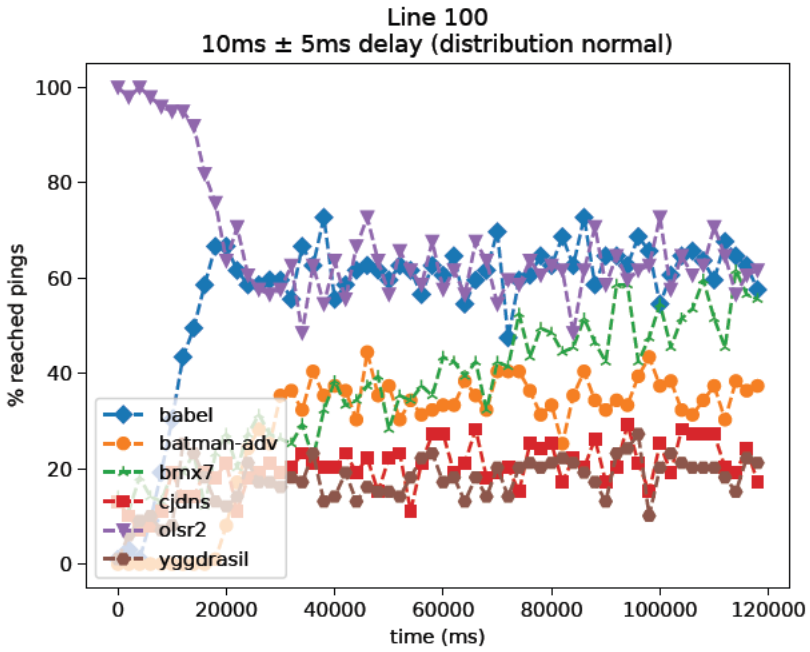


Figure 4.8: Environment with fluctuating latencies and 100 nodes in a line 100 topology

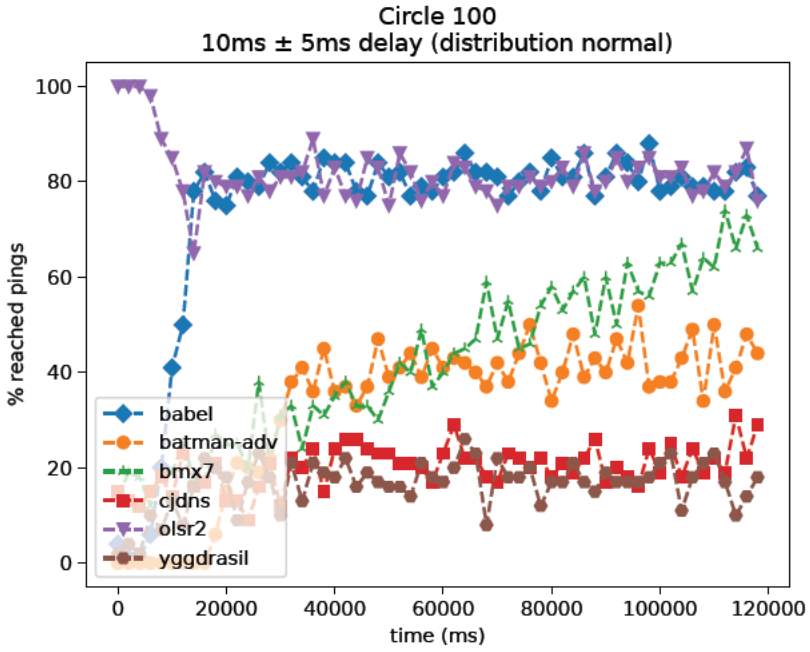


Figure 4.9: Environment with fluctuating latencies and 100 nodes in a circle 100 topology

4.3.3 Packet loss 2 percent

In an environment where each link has a randomized 2 % packet loss, yggdrasil handles this packet loss better than the competitors. This can be seen in Figure 4.10 and Figure 4.11. It works as an overlay network, this could encapsulate the packet loss to the underlying ISO/OSI layer. Further verification of this is needed.

Since not only the routing data is affected by the packet loss, also the pings, slightly unexpected that the pings achieved collapse

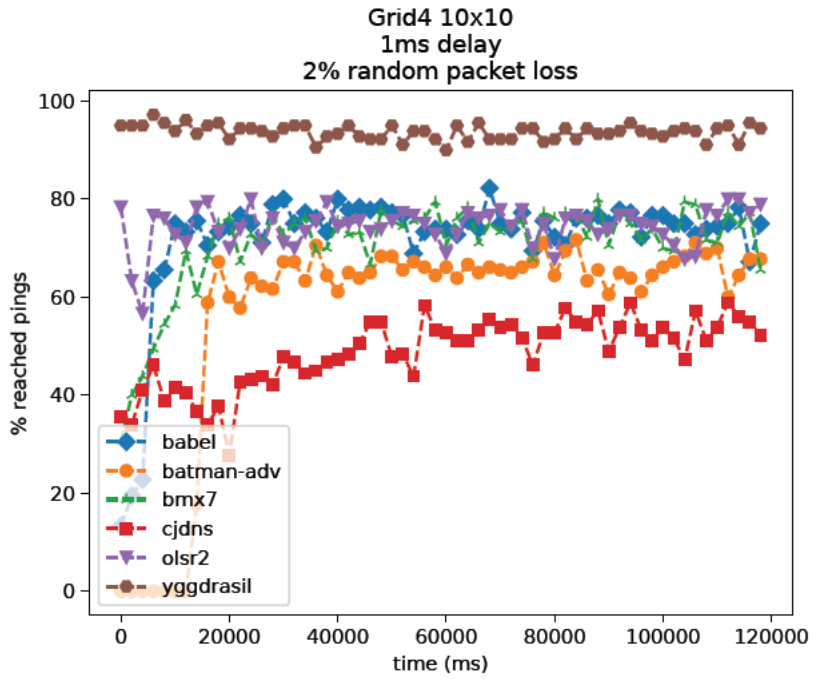


Figure 4.10: Environment with high packet loss on a gird4 10x10 topology

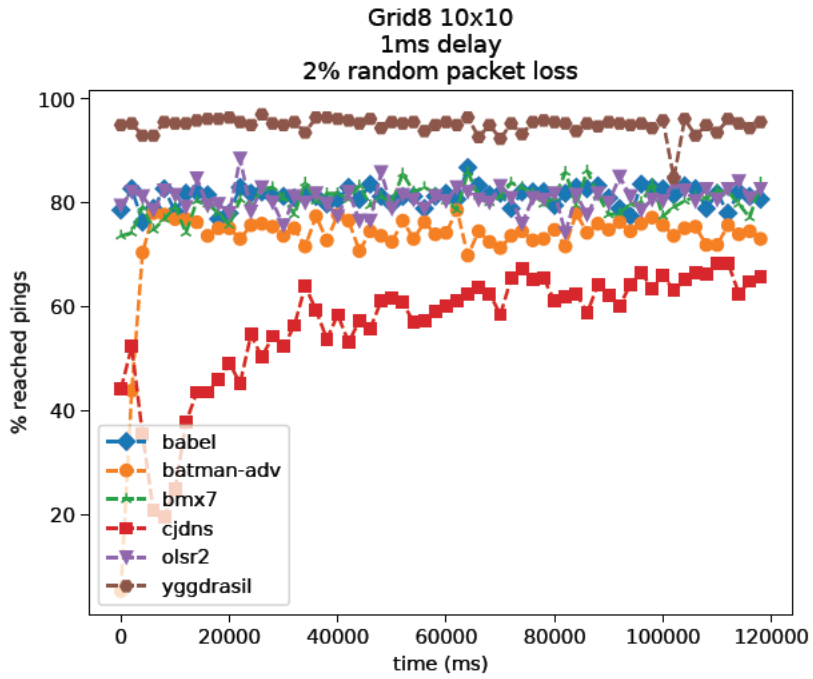


Figure 4.11: Environment with high packet loss on a grid8 10x10 topology

Again, a falloff in convergence was measured for olsr2 in the line (Figure 4.12) and circle (Figure 4.13) topology. Even for the line/circle topologies yggdrasil can reach a convergence about 80 %, which is more than the other competitors.

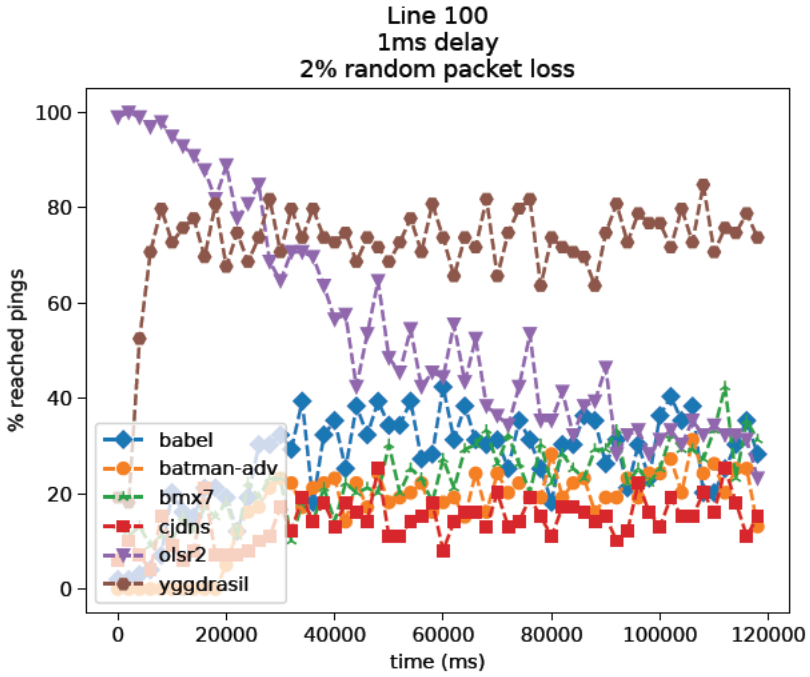


Figure 4.12: Environment with high packet loss on a line 100 nodes topology

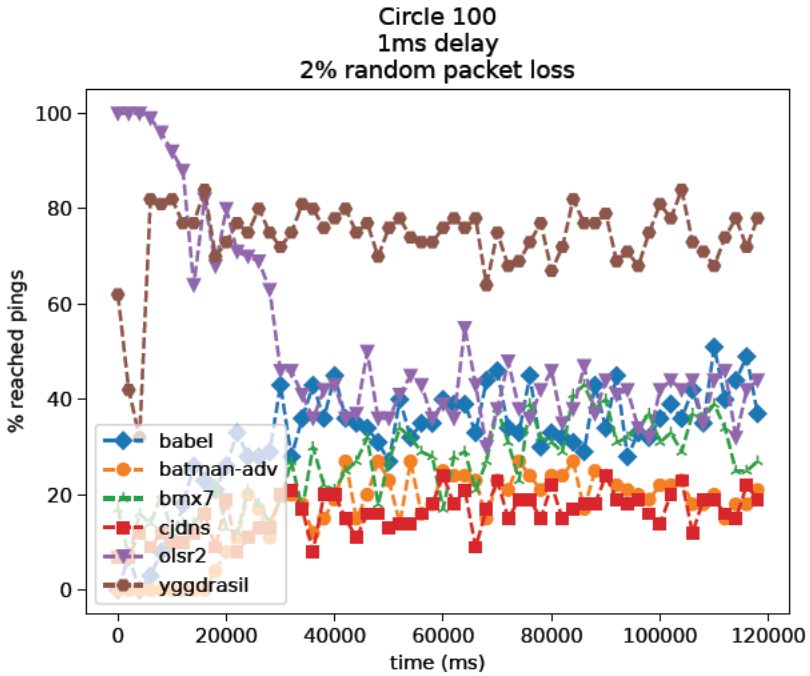


Figure 4.13: Environment with high packet loss on a circle 100 nodes topology

4.3.4 Delay with 10ms ± 5ms and 0.2 percent packet loss

In an environment with fluctuating latencies 10 ± 5 ms (normal distribution) and where each link has a randomized 0.2 % packet loss, olsr2 and babel are the first to reach 100% convergence in the grid (Figure 4.14 and Figure 4.15) topologies. Batman-adv and bmx7 also quickly obtain a high convergence. Interestingly, yggdrasil and cjdns are far behind as shown in Figure 4.6 and Figure 4.7. The graphs have a bit more noise than the other graphs (Figure 4.6 and Figure 4.7) without packet loss.

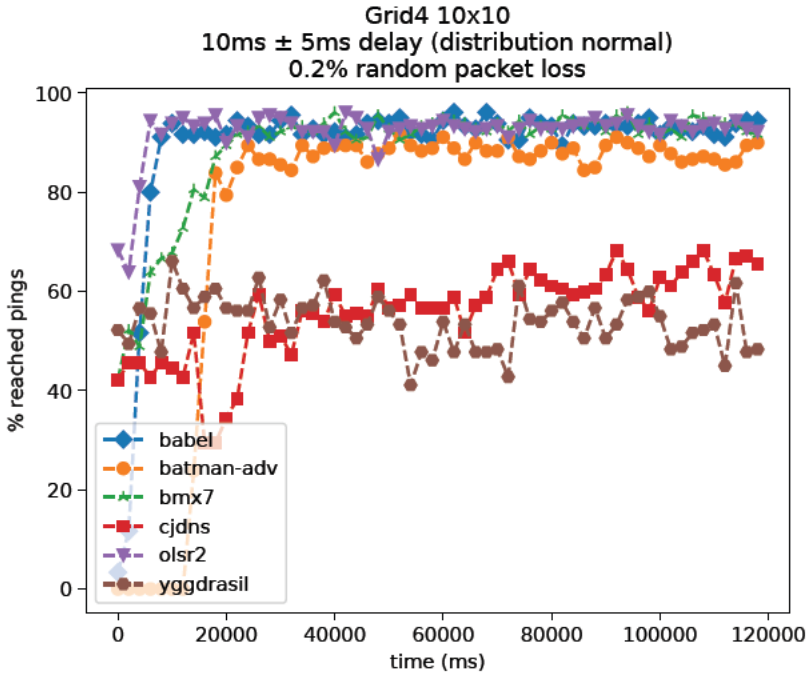


Figure 4.14: Environment with few packet losses and fluctuating delay on with a grid4 10x10 topology

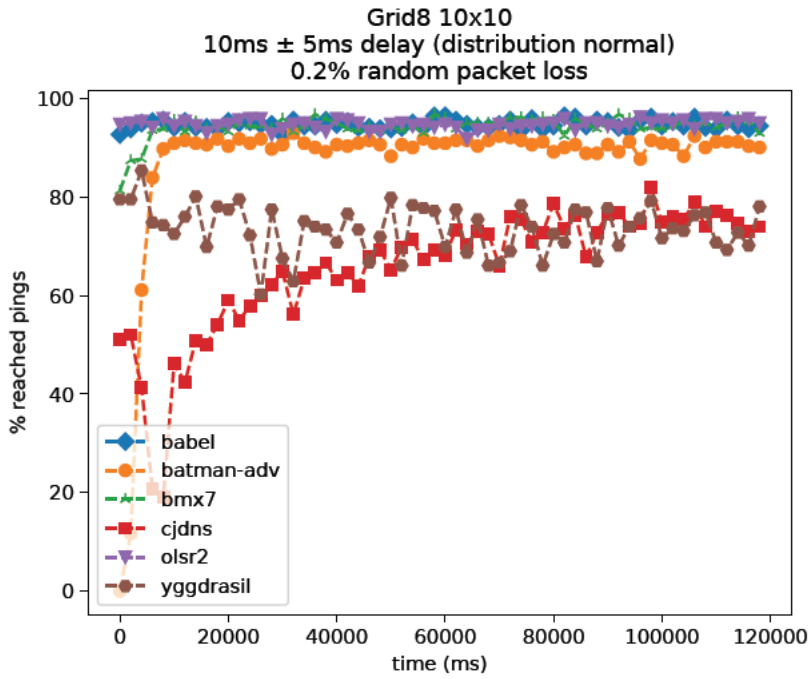


Figure 4.15: Environment with few packet losses and fluctuating delay on with a gird8 10x10 topology

The line Figure 4.16 and cirlce topologie (Figure 4.17) have a similar behaviour as the grid topologies. Compared to the Figure 4.8 and Figure 4.9 topologies, the graphs are nearly identically, just added a bit more noise between each run.

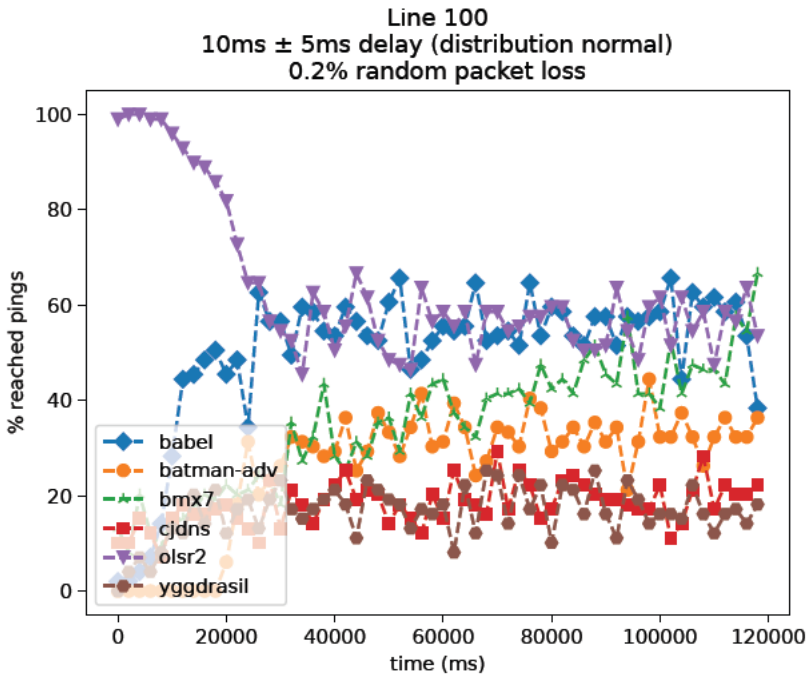


Figure 4.16: Environment with few packet losses and fluctuating delay on with a line 100 nodes topology

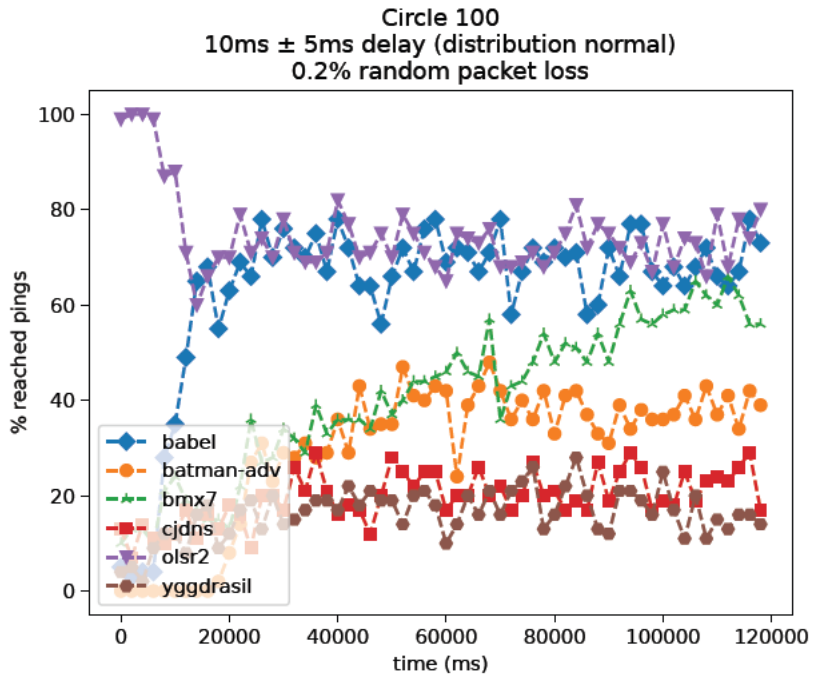


Figure 4.17: Environment with few packet losses and fluctuating delay on with a circle 100 nodes topology

4.3.5 Delay with 10ms ± 5ms and 2 percent packet loss

In an environment where each link has randomized 2 % packet loss and with fluctuating latencies 10 ± 5 ms (normal distribution). The protocols have the weakest results in the Figure 4.18, Figure 4.19, Figure 4.20 and Figure 4.21). However, due to the most difficult network conditions, this was to be expected.

Babel, bmx7 and olsr2 have a similar behaviour as the grid topologies and performs better than their competitors.

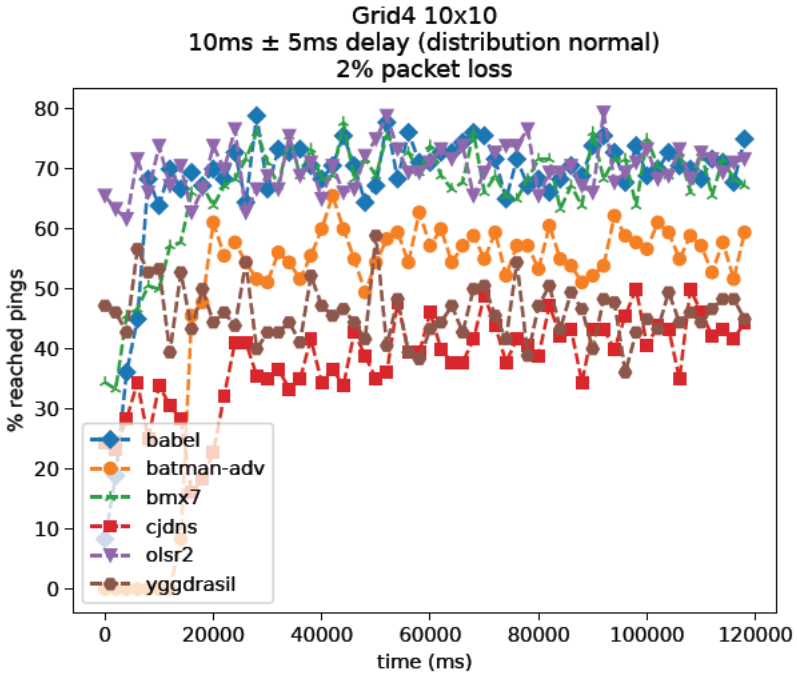


Figure 4.18: Environment with packet loss and delay on a grid4 10x10 topology

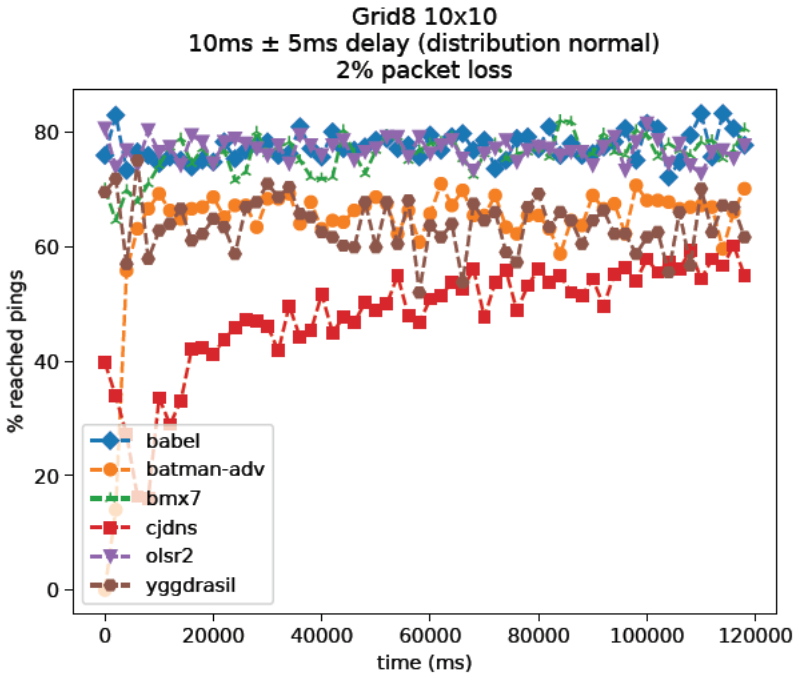


Figure 4.19: Environment with packet loss and delay on a grid8 10x10 topology

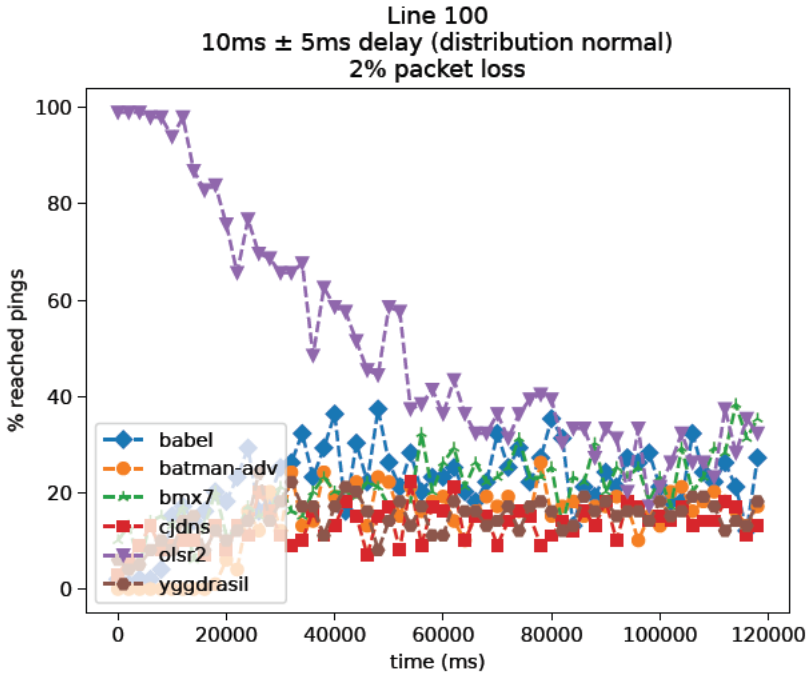


Figure 4.20: Environment with packet loss and delay on a line 100 topology

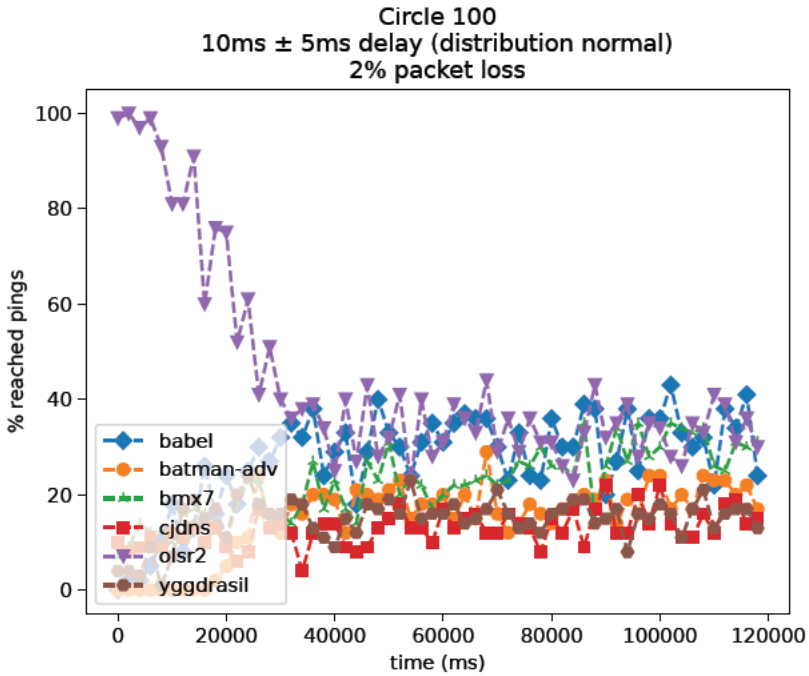


Figure 4.21: Environment with packet loss and delay on a line 100 topology

4.3.6 Summary of convergence

Due to the hop limits in some protocols, it was not possible to perform a measurement with tc between all node pairs. These were labelled with did not finish (DNF).

Babel, olsr2, maybe bmx7 demonstrated the best performance in the convergence analysis. Batman-adv has often a warm up time before it can reach other nodes. Cjdns and yggdrasil seem to have big difficulties on networks with fluctuating latencies.

4.4 Bandwidth Analysis

Each run is performed three times for each topology. The links were set to 100 Mbit/s. The average is of all three runs and rx/tx directions are taken into account. The topologies from Topologies with 100 nodes are used.

In Table 4.2, Table 4.3 and Table 4.5 the bandwidth not affected by topologies (line, grid4 and grid8) for bmx7, olsr2 and babel. The difference between the three protocols is so small. This can be treated as measurement inaccuracies. Yggdrasil has about 5 % to 10 % less bandwidth. Olsr2 has about 50 % to 60 % less bandwidth. Batman-adv has the weak performance across the 3 topologies. This may be caused by fragmentation, as mentioned on the project website[2].

Nodes	0-9	0-55	0-90	0-99
babel	**93.6 Mbit/s**	93.6 Mbit/s	93.5 Mbit/s	**93.4 Mbit/s**
batman-adv	15.7 Mbit/s	8.1 Mbit/s	15.8 Mbit/s	4.5 Mbit/s
bmx7	93.5 Mbit/s	**94.2 Mbit/s**	93.6 Mbit/s	92.6 Mbit/s
cjdns	48.6 Mbit/s	37.7 Mbit/s	37.0 Mbit/s	DNF
olsr2	**93.6 Mbit/s**	93.5 Mbit/s	**93.7 Mbit/s**	92.9 Mbit/s
yggdrasil	85.1 Mbit/s	86.7 Mbit/s	65.2 Mbit/s	72.1 Mbit/s

Table 4.2: Bandwidth measured in a grid4 topology with 100 nodes

In Bandwidth measured in a line topology with 100 nodes the bandwidth is affected by topology (line) for every MANET protocol. Babel, bmx7 and olsr2 are has the most bandwidth in this topology. A measurement between the nodes 0-16 and 0-32 has nearly no impact on the protocols. The measurements between the nodes 0-64 and 0-99 has a

[2] https://www.open-mesh.org/projects/batman-adv/wiki/Quick-start-guide, accessed March 2022

Nodes	0-9	0-55	0-90	0-99
babel	**93.6 Mbit/s**	93.6 Mbit/s	93.6 Mbit/s	**93.5 Mbit/s**
batman-adv	1.6 Mbit/s	2.7 Mbit/s	1.6 Mbit/s	1.5 Mbit/s
bmx7	93.3 Mbit/s	**93.7 Mbit/s**	93.6 Mbit/s	93.4 Mbit/s
cjdns	29.2 Mbit/s	30.2 Mbit/s	34.0 Mbit/s	DNF
olsr2	**93.6 Mbit/s**	93.6 Mbit/s	**93.7 Mbit/s**	**93.5 Mbit/s**
yggdrasil	85.7 Mbit/s	89.2 Mbit/s	86.7 Mbit/s	85.5 Mbit/s

Table 4.3: Bandwidth measured in a grid8 topology with 100 nodes

Nodes	0-16	0-32	0-49	0-64
babel	92.8 Mbit/s	92.1 Mbit/s	**90.3 Mbit/s**	90.2 Mbit/s
batman-adv	31.5 Mbit/s	18.0 Mbit/s	DNF	DNF
bmx7	89.3 Mbit/s	**92.6 Mbit/s**	88.8 Mbit/s	90.9 Mbit/s
cjdns	DNF	DNF	DNF	DNF
olsr2	**93.3 Mbit/s**	90.4 Mbit/s	89.9 Mbit/s	**92.3 Mbit/s**
yggdrasil	73.0 Mbit/s	43.1 Mbit/s	30.2 Mbit/s	23.7 Mbit/s

Table 4.4: Bandwidth measured in a circle topology with 100 nodes

bandwidth reduction about 25 % to 50 % for babel and olsr2. Yggdrasil and batman-adv loses bandwidth with increasing distances.

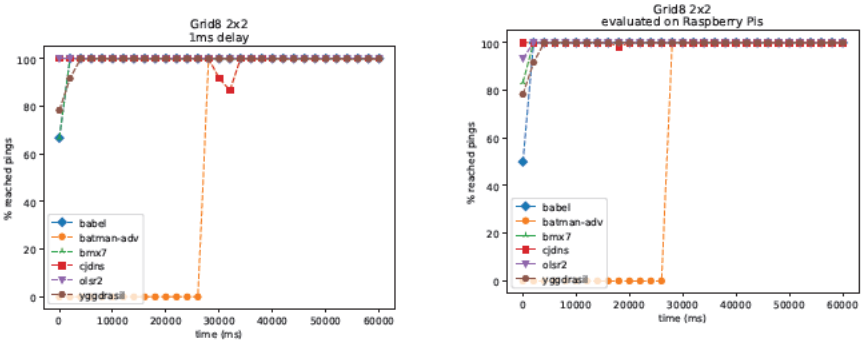
Nodes	0-16	0-32	0-64	0-99
babel	92.7 Mbit/s	92.1 Mbit/s	**76.9 Mbit/s**	50.4 Mbit/s
batman-adv	31.8 Mbit/s	18.0 Mbit/s	DNF	DNF
bmx7	**93.0 Mbit/s**	92.5 Mbit/s	DNF	DNF
cjdns	DNF	DNF	DNF	DNF
olsr2	89.7 Mbit/s	**92.3 Mbit/s**	73.3 Mbit/s	**53.3 Mbit/s**
yggdrasil	72.0 Mbit/s	43.3 Mbit/s	23.6 Mbit/s	15.7 Mbit/s

Table 4.5: Bandwidth measured in a line topology with 100 nodes

4.5 Validation

As expected, such a small network is very hard to measure. As shown in Figure 4.22, every protocol converts so rapidly, it is almost not measurable. It is noticeable that the batman-adv protocol has a point at the beginning where no nodes are reachable. Similar behavior is recognizable in Figure 4.2.

All this is an indication that even larger emulations lead to a result that is valid.



(a) Emulation a grid8 topology with 2x2 nodes

(b) Validation Setup with Raspberry Pis connected like a grid8 topology with 2x2 nodes

Figure 4.22: Validation Results with an emulated grid8 and a Raspberry Pi with grid8 topology with 2x2 nodes

5 Conclusion

In the Loomo scenario with 4 nodes, the choice of the MANET protocol used is not really important. All protocols achieve good results in convergence time which can be seen in Figure 4.22. In the prospect when fleet size is increased, babel is recommended. Babel handles all 4 topologies in each scenario very well, however olsr2 gave almost the similar results. Due to the inconsistencies of the convergence behaviour of olsr2, in the line and circle topologies, this should be furthered separately investigated in a work.

For Babel, bmx7 and olsr2 had reached nearly the full theoretical bandwidth. The results were very close, any differences in tolerances can arrogate to a measurement inaccuracy.

Babel outperformed its competitors in Convergence Analysis and Bandwidth Analysis, which is recommended for the given Loomo scenario. If possible, there should be several links between the nodes, like a grid4 or grid8. Convergence in these networks was faster than in networks with fewer links among them (e.g. line or cricle topologies).

Finally, it would be recommended in a future work to extend the experiment by changing the topology while evaluating and a validation of the emulations with 100 nodes to compare to real world systems. Instead of pings for convergence analysis, it is possible to evaluate the routing tables of all nodes. This should lead to more reliable results in the case of packet loss.

Bibliography

[1] A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*. USA: Prentice Hall Press, 5th ed., 2010.

[2] S. Eichler, C. Schroth, and J. Eberspächer, "Car-to-car communication," in *Proceedings of the VDE-Kongress - Innovations for Europe*, p. 6, VDE Verlag, Oktober 2006.

[3] S. S. Jadhav, A. V. Kulkarni, and R. Menon, "Mobile ad-hoc network (manet) for disaster management," in *2014 Eleventh International Conference on Wireless and Optical Communications Networks (WOCN)*, pp. 1–5, 2014.

[4] D. S. M. Corson and J. P. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations." RFC 2501, Jan. 1999.

[5] W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[6] L. Yang, P. Zerfos, and E. Sadot, "Architecture Taxonomy for Control and Provisioning of Wireless Access Points (CAPWAP)." RFC 4118, June 2005.

[7] Contributors, "Babel is a loop-avoiding distance-vector routing protocol for ipv6 and ipv4 with fast convergence properties. it is based on the ideas in dsdv, aodv and cisco's eigrp, but is designed to work well not only in wired networks but also in wireless mesh networks, and has been extended with support for overlay networks. babel is an ietf standard.."

[8] J. Chroboczek and D. Schinazi, "The Babel Routing Protocol." RFC 8966, Jan. 2021.

[9] Contributors, "B.a.t.m.a.n. (better approach to mobile ad-hoc networking) is a routing protocol for multi-hop ad-hoc mesh networks."

[10] Contributors, "Bmx7 is a mesh routing protocol for linux based operating systems.."

[11] A. Neumann, L. Navarro, and L. Cerdà-Alabern, "Enabling individually entrusted routing security for open and decentralized community networks," *Ad Hoc Networks*, vol. 79, pp. 20–42, 2018.

[12] Contributors, "The batman project that started with a basic protocol and a single userspace routing daemon, attracted quite some attention over the years which led to many new ideas and concepts meant to improve the project.."

[13] Contributors, "olsrd and olsrd2 are both link state routing protocol implementations optimized for mobile ad hoc networks on embedded devices like commercial of the shelf routers, smartphones or normal computers."

[14] T. H. Clausen, C. Dearlove, P. Jacquet, and U. Herberg, "The Optimized Link State Routing Protocol Version 2." RFC 7181, Apr. 2014.

[15] Contributors, "Cjdns implements an encrypted ipv6 network using public-key cryptography for address allocation and a distributed hash table for routing. this provides near-zero-configuration networking, and prevents many of the security and scalability issues that plague existing networks.."

[16] Contributors, "Yggdrasil is an early-stage implementation of a fully end-to-end encrypted ipv6 network. it is lightweight, self-arranging, supported on multiple platforms, and allows pretty much any ipv6-capable application to communicate securely with other yggdrasil nodes.."

[17] M. Zhang, H. Wen, and J. Hu, "Spanning Tree Protocol (STP) Application of the Inter-Chassis Communication Protocol (ICCP)." RFC 7727, Jan. 2016.

[18] "Internet Protocol." RFC 791, Sept. 1981.

[19] D. S. E. Deering and B. Hinden, "Internet Protocol, Version 6 (IPv6) Specification." RFC 8200, July 2017.

Glossary

Bluetooth Bluetooth is a wireless personal area network protocol..

CaDS group A working group at the HAW Hamburg that focuses on distributed systems and general networking..

hop A hop occurs when a packet is forwarded from one node to another..

Loomo The Segway Loomo Robot is an advanced personal smart vehicle..

LoRaWAN Long Range Wide Area Network (LoRaWAN) is a wireless personal area network protocol..

mesh network A mesh network is a network of connected nodes that can communicate with each other without a central server. Typcially, every node connects to many nodes as possible..

Raspberry Pi A small single-board Advanced RISC Machines (ARM) computer.

Thread Thread is a wireless personal area network protocol..

Tor Network The Tor network is an overlay network that works in a decentralized manner and has as its main purpose to anonymize its users..

ZigBee ZigBee is a wireless personal area network protocol..

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original