Robustness and Scalability Improvements for Distance Vector Routing in Large WMNs

Martin Backhaus and Markus Theil and Michael Rossberg and Guenter Schaefer
Technische Universität Ilmenau, Germany
[martin.backhaus, markus.theil, michael.rossberg, guenter.schaefer][at]tu-ilmenau.de

***Abstract*—IEEE 802.11s enables rapid deployment of Wireless Mesh Networks (WMNs) to supply basic wireless connectivity for client hardware. Application of distance vector based routing protocols proved advantageous in WMNs for efficiency, i.e., low overhead. However, it remains unclear, if plain distance vector routing protocols allow for adequate robustness against outages in large installations. Particularly, the required time for routing re-convergence may be too long, as in practice, the count-to-infinity phenomenon needs to be dealt with. This paper proposes Spare Forwarding Entries (SFEs), resulting in a proactive mechanism improving robustness against outages when compared to the reactive behavior of distance vector protocols. It works as an extension of distance vector operation, i.e., efficiency can be preserved. We integrate SFE into the well-known Babel routing protocol and also propose measures to increase scalability. Simulation studies indicate that SFEs improve robustness against node outages significantly. In certain scenarios, the Packet Delivery Ratio (PDR) was doubled with our mechanisms in place. Further modifications let Babel-based WMNs scale up to 300 nodes. Moreover, convergence times and overhead are significantly smaller.**

***Index Terms*—Wireless Mesh Networks, Distance Vector Routing, Robustness**

I. INTRODUCTION

Wireless Mesh Networks (WMNs) have been an active research area for many years, but there is a huge discrepancy between systems in research publications and approaches using current IEEE 802.11 compliant hard- and software that actually made it into practice. Robust, large installations equipped with many mesh routers are still infeasible due to deficiencies regarding robustness and scalability [1]. For certain use cases, robustness is paramount when providing wireless connectivity, e.g., in first responder scenarios. The nature of WMNs requires a fully distributed operation of all essential protocol mechanisms. Therefore, mesh solutions from companies like Cisco or Aruba (which employ Wi-Fi controllers and Layer 2 tunnels) are rendered impractical, as the controller depicts a Single Point of Failure (SPOF). Even if the controller itself does not fail or if it has a backup, paths to the controller carry much more load than necessary and depict a bottleneck (all traffic traverses the controller). Hence, routing needs to be tackled in a distributed fashion and most likely on Layer 3.

Scalability issues in distributed WMNs originate not only in the stochastic channel access of Wi-Fi, which is sensitive to signal interference, but also from broadcast traffic to distribute routing information in most routing protocols. While offering overhearing advantage and learning from messages destined for other nodes, broadcast in Wi-Fi networks uses low transmission rates and therefore requires a significant amount of airtime.

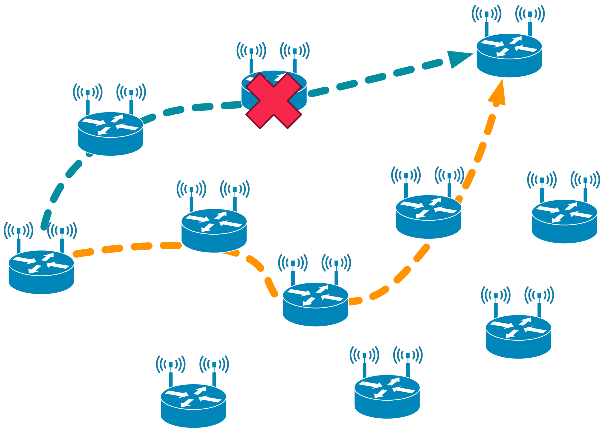

Fig. 1. Use Case of Babel-SFE.

Existing open mesh routing protocols like OLSR [2] or B.A.T.M.A.N. [3] provide no means to explicitly provide high robustness, i.e., fast and proactive route repair mechanisms. Approaches based on distance vector routing have already been identified to be more scalable [4], but their robustness relies on re-convergence of routes. Depending on how the count-to-infinity phenomenon (inherent to distance vector approaches) is solved, re-convergence of lost routes may take unacceptably long. If the distance vector paradigm could be revisited and this inherent issue alleviated, it's acceptance for real-world applications in WMNs may greatly improve. Scalability enhancements would further contribute to acceptance, especially when dealing with lots of nodes participating in a WMN.

Fig. 1 shows a scenario for a WMN with a possibly large number of routers. These networks might be used in first responder scenarios or large industrial/commercial installations and need to provide network connectivity despite temporary node outages. As illustrated, proactive measures would be able to immediately regain connectivity after node outages.

Given the sketched deficits, this paper proposes a mechanism called Spare Forwarding Entry (SFE) to significantly improve the robustness of distributed distance vector routing algorithms. In addition to the forwarding entries already established by usual distance vector operation, SFE proactively creates and maintains another set of forwarding entries, which can be used

---

This work was supported by RWTÜV-Stiftung (Essen, Germany) under grant number S0189/10033/2018.

in case of outages. This way, connectivity can be immediately restored without the need for routing re-convergence.

We decided to use the well-established Babel routing protocol [5] as a basis and provide enhancements for scalability as well and to systematically analyze, enhance, and fine-tune the approach for large WMN installations. Relying on state-of-the-art protocols should be preferable over designing a completely new routing protocol, because Babel already provides a good starting point and has several advantages over other existing WMN routing protocols. Babel already employs distance vector mechanisms combined with an efficient update mechanism, which does not rely on flooding information through the entire network. Furthermore, it is flexible due to usage of Type Length Values (TLVs) and already provides built-in hooks for extension. However, when using the recommended operation parameters, Babel is rather chatty as it periodically broadcasts update messages. Proactive robustness mechanisms for storing additional backup paths are neither built into Babel, nor in any other state-of-the-art routing protocol based on distance vector operation.

In particular, we provide the following contributions:

1) Devising and integrating SFE for backup paths to enhance WMN robustness
2) Enhanced scalability (reliable transport of routing packets)
3) A comprehensive evaluation using simulation studies

The remainder of this paper is organized as follows: First, we motivate requirements for robust routing in large WMNs and provide an overview of state-of-the-art routing protocols and approaches (Sec. II). We present our approach in Sec. III and explain relevant details. The evaluation in Sec. IV discusses our approach qualitatively and studies its behavior in case of outages, initial convergence time, and overhead. To sum up, Sec. V presents some concluding thoughts and directions for future research.

II. REQUIREMENTS AND RELATED WORK

Based on several shortcomings of the current IEEE 802.11s mesh implementation (see [1] for details), we identify crucial requirements. To work in real-life environments, a WMN has to fulfill the following main functional requirements:

- Capability to proactively deal with link and node failures (uncorrelated)
- No SPOF (like Wi-Fi controllers)

Far more important for large installations are the following quantitative requirements:

- High Packet Delivery Ratio (PDR) despite outages
- Scalable mesh routing protocol (usable with > 150 nodes)

Concerning state-of-the-art approaches, AODV-MBR [6] is based on Ad hoc On-Demand Distance Vector Routing (AODV) and uses additional backup routing entries, which are learned from routing messages to neighbors in promiscuous mode. This approach is an extension which works without additional routing traffic for backup routes, but suffers from the same high overhead as AODV and does not scale well. We mention this approach explicitly to avoid confusion, as the name may suggest proactive mechanisms for robustness (backup paths) in a distance vector routing protocol. But AODV itself is not a proactive distance vector protocol – it rather relies on path request/response dialogs to establish end-to-end paths on demand.

BMX6 [7] evolved from B.A.T.M.A.N. and combines ideas found in B.A.T.M.A.N. and Babel. It aggregates Originator Gateway Messages (OGMs) and redistributes them only if necessary, but still relies on periodic broadcast of OGMs on so-called relevant links. Please note that the more recent BMX7 [8] is mainly a security enhancement to BMX6 and not discussed in this paper. While BMX6 and BMX7 provide scalability improvements over B.A.T.M.A.N., they do not add additional resilience.

EIGRP [9], although not described as such, has the concept of *feasible successors* which implicitly resembles backup forwarding entries. They are chosen by handling a neighbor's announced metric and resulting metric (when incorporating link costs to the neighbor) differently and thereby deriving forwarding entries not having the smallest metric value. This procedure will, however, not guarantee a backup forwarding entry, although one could exist. EIGRP does not provide explicit mechanisms to establish feasible (loop-free) backup entries. Furthermore, it is designed for wired communication and therefore not suited for frequent metric changes and mobility events known from wireless communication.

Distance vector algorithms in their basic form execute a distributed Bellman-Ford algorithm, yielding shortest paths to all destinations by exchanging neighbor information and metrics, i.e., distance vectors. As widely known, this approach is scalable and fast in the best case, but suffers from the count-to-infinity problem. Solving the count-to-infinity problem is possible with additional restrictions or invariants on route updates, see Diffusing Update Algorithm (DUAL) [10], Loop-free Invariant (LFI) [11] or Distributed Path Computation with Intermediate Variables (DIV) [12]. They all try to apply updates guaranteeing loop-freedom at every point in time.

Babel [5] is a distance vector routing protocol, using the idea of feasibility constraints based on sequenced routes to avoid routing loops. Route distribution is achieved by periodic updates. Dead Peer Detection (DPD) uses periodical Hello and I Heard You (IHU) messages. Every Babel speaker is identified using a 64 Bit router ID to unambiguously identify sources of announcements, even if an IP prefix is being announced from multiple mesh routers. While Babel is already rather scalable because of its distance vector approach, it lacks proactive mechanisms for increased robustness. In case of node outages, Babel speakers will most likely experience route changes: Metrics will become more expensive and next hops may change. In this case, Babel would not simply adopt a route, because a more expensive metric violates feasibility constraints and requires explicit and reactive measures to resolve, i.e., sequence number requests. As a reactive measure, they require one complete round trip to the prefix' source to regain connectivity, which could only be improved by proactive mechanisms.

As long as enhancements to distance vector routing use only a constant amount of knowledge/storage per node, distance vector algorithms can keep their scalability property. Equal cost multi-path routing is possible, when storing all neighbors with the same minimal distance towards destinations as next hops [13]. Another possible enhancement is a combination of network wide distance vector routing with a more complete link-state-like view on the local neighborhood [14]. Such an approach can be used for multi-path routing or local repair mechanism in case of link failures. Yet another approach [15] stores multiple local spanning tree fragments using the distance vector information. The incoming link is used to distinguish, if a packet uses an optimal or non-optimal (backup) route. In the non-optimal case, using alternate paths can be continued.

We considered robust routing in WMNs [16] using interference disjoint backup-paths. However, neither did the approach emphasize scalability aspects, nor was it based on efficient distance vector operation. Nonetheless, it contains a detailed state-of-the-art analysis in terms of WMN robustness.

Main Takeaways

Summing up on state of the art, there is no protocol that is both robust and scalable. All in all, most algorithms use loop-free constraints, enriched or multiple distance vectors, and base forwarding decision on incoming links or unreachable links in the short-term past. Babel is the most promising routing protocol in terms of scalability [4]. It is versatile through TLV extensions and is therefore chosen as the basis for our own work. To the best of our knowledge, this article is the first one to present proactive robustness mechanisms within a distance-vector based protocol in a WMN context.

III. BABEL-SFE: INTEGRATING SPARE FORWARDING AND IMPROVING SCALABILITY

Our enhancements to Babel, called Babel-SFE (short for Spare Forwarding Entry), mainly consist of two parts: 1) *Robustness enhancements* using a novel mechanism of SFE, which depicts backup forwarding entries that can be chosen in case of an outage. 2) *Reduction of routing overhead* by disabling periodic updates and employing reliable transport, which in turn enhances routing scalability. The following subsections discuss these parts in detail.

A. Increasing Robustness with Spare Forwarding Entries (SFEs)

Mainly, the idea of SFE is similar to backup paths from other, well-known routing protocols. The main difference is that we create backup forwarding entries using a regular distance-vector operation (of Babel in this case), relying on local knowledge, only. The backup forwarding entries (or SFEs) should achieve path diversity when compared to the main forwarding entry if used in case of outages. Most importantly, updates for SFEs can be handled in the same way as regular Babel updates, therefore, loop avoidance is automatically accomplished. We try to achieve the desired diversity of SFEs by simple metric modifications to keep additional overhead small. The main idea of metric adjustments is to add a penalty to the metric of spare routes for links, if this link is already used for regular forwarding. Using these penalties, the spare route metric will create paths which are diverse to the ones formed by regular forwarding entries if possible.

1) Distribution of Spare Entries: As mentioned before, spare updates will still contain a sequence number and a metric, so they can be handled in the same fashion as regular Babel updates concerning feasibility and necessity of redistribution. Information derived from spare updates are stored in separate data structures, but following the same Babel rules, i.e., a route table to accommodate for spare routes advertised by neighbors and a forwarding table dedicated to SFEs (analogous to regular routes and forwarding entries). Updates for spare routes will be proactively distributed 5 s after the regular route was announced for the first time.

Theoretically speaking, when a node announces a prefix regularly, a reverse routing tree with the announcing node as root is formed. Additional spare route announcements should then construct a second tree, which is preferably as diverse as possible to the first one. To achieve this, we introduce the following additions to the protocol's behavior:

Behavior of the Source: Every source of a prefix will announce spare updates additionally to regular routes. To distinguish them from regular updates, a bit inside the update TLV is set to one, indicating a spare update. Analogously, a zero bit indicates a regular update. Remaining components of the update, i.e., metric, router ID, sequence number, are chosen by the source in exactly the same way as for regular updates. Spare updates will be sent to all Babel neighbors.

Behavior of Other Nodes: All nodes in the network can easily differentiate regular updates and spare forwarding updates by checking the aforementioned bit. Regular updates are treated in the same fashion as in unmodified Babel. If it is a spare forwarding update, the behavior depends on whether or not the update is received from a neighbor, which is used for regular forwarding. In case the neighbor transmitting the spare update *is used for forwarding of regular traffic*, the update must not be used for an SFE. As the next hop would equal the regular forwarding's next hop, this spare route offers no additional diversity or robustness on a local scale. Because it may be necessary to achieve diversity beyond local neighborhood, this update needs to be redistributed. However, as the sending neighbor equals the neighbor used for regular forwarding, a metric penalty is added onto the update's metric value prior to redistribution, to accommodate for usage of the same edge in the network graph for both regular update and spare update. In case the neighbor transmitting the spare update *is not used for forwarding of regular traffic*, the update can be used for an SFE, as its next hop would lead to a different route in case the regular forwarding target breaks down. If two spare routes have the same metric, the regular route metric is used to obtain a decision – leading to a shorter spare route. These spare routes can then be redistributed without any further restrictions.

2) Behavior in Case of an Outage: For the proposed mechanism of SFEs, nodes detecting an outage execute additional protocol instructions. All other nodes use their previously collected spare entries for affected routes if required.

Behavior of the Node Detecting the Outage: If a node detects an outage of a neighbor used for regular forwarding, it will send route retractions to all its remaining neighbors (as usual for Babel). Additionally, it examines the corresponding SFE. The neighbor that is set in this SFE needs to receive the retraction with a bit (set to one) indicating that this neighbor will be used for spare forwarding. This measure will be relevant for the propagation of the breakdown using subsequently sent retractions to decide whether or not to use the spare entries and to prevent the formation of routing loops. Finally, the node will install the SFE as a regular forwarding entry. Please note that because a retraction with the bit indicating the neighbor for future forwarding was set, installation of the SFE should be postponed for a short period of time, e.g., 250 ms, to prevent the formation of temporary routing loops. Furthermore, the node must reliably send the retraction to the receiving node.

Behavior of all Other Nodes: A node receiving a route retraction from a neighbor that is used for regular forwarding, first checks the bit indicating if the node will be used as SFE of the neighbor sending the retraction or not. If this bit is set, the node processing the retraction is vital for the formation of a spare route through the network. In this case, the node's behavior is identical to the one detecting the outage, i.e., sending retractions and installing the SFE. If this bit is not set, the node may use its spare entry if advantageous. This freedom of choice exists, because the retraction-sending node already installed an SFE, i.e., a spare route already exists. That means, the node used for regular forwarding must not be changed, but using the local SFE may lead to a route with a better metric value with respect to the spare route's metric. Consequently, whether or not the SFE is installed depends on metrics announced for spare routes from different neighbors, which are stored in the route table dedicated for spare routes. If the spare route identified by the node's SFE has a smaller spare metric than the spare route offered by the retraction-sending node, an SFE is installed and retractions sent as said before.

3) Notes on Integration: It is worth mentioning that there is no guarantee for a spare route to exist and that it will actually lead to a diverse path. In our testing, however, we were able to find a spare route with extremely high probability, thereby alleviating the simulated outage. More effort on realizing comprehensive Babel integration is still necessary to incorporate route requests and sequence number requests for spare routes as well, prior to productive use to provide long-term maintenance of routes according to Babel's principles. However, this has no significant implications for robustness evaluation performed in this paper. Furthermore, there are no fundamental limitations to be expected when implementing remaining features to finalize the integration.

B. *Overhead Reduction and Fast Convergence*

Babel in its original form uses soft state and hence requires periodic updates, even if there are no changes in the network. Our modified Babel omits periodic updates and guarantees message reception by using an ACK mechanism to retransmit if necessary, therefore providing reliable transport of routing protocol messages. Updates are only sent in case of topology changes or if they were requested with Babel sequence number requests or route requests, e.g., occurring in case of new mesh connections. Changes touching client routes cause urgent/triggered updates to be sent for fast convergence. To achieve robustness without relying on simple timeouts, DPD and according route retractions are employed. TLVs for which an ACK is outstanding are aggregated. As usual for Babel, an aggregation mechanism tries to send as many TLVs as possible to send large packets instead of multiple small ones, thereby saving airtime. Please note that the Babel RFC [5] also specifies TLVs for acknowledgement requests and acknowledgments, but they are not employed in Babel's reference implementation [17]. We examine the importance of reliable transport for network convergence over the course of our quantitative evaluation.

IV. EVALUATION

We evaluated Babel-SFE using simulation studies. This section presents scenarios, experiments and performance metrics of our evaluation in detail. We discuss our mechanisms qualitatively first, before giving quantitative results averaged over 32 repetitions with 99% confidence intervals.

A. *Qualitative Discussion*

Our developed approaches meet initially stated functional requirements. Node and link failures are overcome using the SFE mechanism. As we will show in the quantitative evaluation, we reduced the routing overhead significantly to improve scalability of Babel-SFE when compared to regular Babel. None of the presented mechanisms requires a centralized instance, therefore, no SPOF exists.

SFE Mechanism: Establishment of SFEs results in a proactive mechanism that does not create additional traffic in case of outages, but prior to failure conditions – avoiding additional overhead in critical situations. Each spare route is serialized using the regular route format and therefore using the same size in Bytes – only employing a tailored metric to suit the objective of creating spare routes. Usage of Babel TLVs leads to a clean integration into the protocol. Because TLVs are aggregated and spare route updates should not be considered urgent or vital, they can be easily aggregated into routing packets that would have been sent anyways (e.g. packets carrying hello or IHU TLVs), therefore leading to no additional overhead in terms of number of packets. Furthermore, when using the SFE mechanism, there is no need to wait for route re-convergence in case of outages, because a spare route that can be used for forwarding is readily available. We will cover this in the quantitative evaluation by examining the PDR in an outage scenario.

B. *Quantitative Results*

A previously developed framework [18] enabling simulation and real-system experiments from one exact codebase is used as foundation of the quantitative evaluation in this work. Since its publication, its simulation component has been modified and extended to work with ns-3 [19], while its real-world

TABLE I
SETTINGS AND EXPERIMENT FACTORS (F)

Setting/Factor	**Values**
Experiment Runs	32 per factor combination
Simulated Time (One Run)	180 s
Standard	IEEE 802.11g
Wi-Fi Mode	Mesh Point & Access Point
Beacon Interval	1000 ms
Startup Interval	5 s
Topologies (F)	Connected Grids, NPART
Topology Sizes (F)	$N \in \{2, 3, 4, 5\}$ (Connected Grids), $50, 100, \dots, 300$ (NPART)
ns-3	3.29

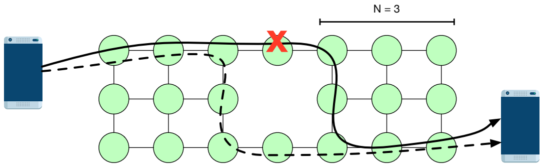

Fig. 2. Connected grid topology (here: $N = 3$) for SFE evaluation.

component has been adapted to use the Linux kernel and its functionalities to realize experiments within an on-site testbed. Our quantitative evaluation in this paper uses simulation studies only, however, a real-system integration will be subject to further studies. Table I summarizes relevant settings, tools, and software versions in all our experiments. For simulative evaluation, we incorporate the NPART topology generator [20] to obtain WMN topologies that resemble real-world WMNs. The topologies will be used for convergence time and overhead measurements. We will conduct experiments starting with 50 and going up to 300 nodes in increments of 50. Another type of topology are two grids of size N which are connected using two nodes – one at the top and one at the bottom of the grids – resulting in a topology depicted in Fig. 2. We refer to these topologies as *Connected Grids*. They will portray a particularly challenging scenario for evaluation of the SFE mechanism in a later experiment. To alleviate issues of simultaneous startup of all nodes at exactly the same time, nodes start uniformly random within the first 5 s of a simulation run.

Simulation Experiment 1 – Node Outage: This experiment investigates the performance of the introduced SFE mechanism. Connected grid topologies (see Fig. 2) are employed in the following fashion: Except for the bottom connecting node (between the grids), all nodes start at the beginning of the simulation (within the first 5 s). By starting the bottom connecting node at 40 s, we ensure that the route between the two clients will traverse the top connecting node. This is because the path traversing the bottom node is not cheaper for a hop-based routing metric (as used for this experiment). Subsequently, spare routes can be distributed in the network. Please note that the connected grid topology is used for SFE evaluation, because it depicts a rather extreme case: Two relatively large parts of the network are connected using two critical nodes. These two nodes are topologically far apart, further aggravating the scenario. The top connecting node will fail at simulation time 90 s. Furthermore, between 90 s and 95 s, the client on the left side will send 500 UDP packets per second to the client on the right. To evaluate the performance of our approach, we will compare the PDR when using SFE against the mechanisms of regular Babel.

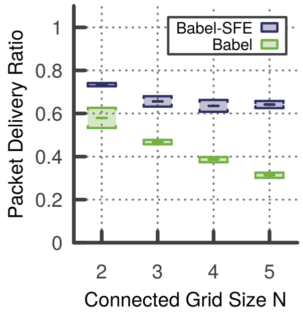

Fig. 3. Packet delivery ratio (PDR) in case of an outage.

Fig. 3 presents the achievable PDR after an outage with SFE from Babel-SFE and the original Babel using connected grids as a topology. As one would expect, the relationship between the size of the connected grid and the PDR is inversely proportional. In case of regular Babel, the PDR develops from 58.0 % for $N = 2$ down to 31.3 % for $N = 5$ – resulting in a severe decrease with growing topologies. This can be explained through Babels mechanisms, which require one full round trip to the prefix' source in order to create a new route which alleviates the outage situation (sequence number requests). Furthermore, note that $N = 5$ results in a topology of 52 nodes and end-to-end paths of 14 hops. Using Babel-SFE and its SFE mechanism, repair of failing routes due to node outages is restricted to an area as small as possible and in particular, does not require end-to-end exchange of routing messages. Therefore, routes are repaired faster and the PDR is a lot higher. Starting with 73.4 % for $N = 2$, it decreases only to 63.5 % for larger connected grids.

Simulation Experiment 2 – Convergence Time and Network Overhead: The next experiment will examine routing convergence time, i.e., how much time is required to establish all shortest routes in the entire network, when each node starts from scratch with an empty routing table. We will compare the case of using reliable transport (Babel-SFE) against periodic updates omitting reliable transport (regular Babel). NPART topologies are the foundation of this experiment, as they resemble real-world WMNs. Metrics of interest are the time required until network convergence is achieved (measured from network startup) and the required message overhead in Megabyte (MB). This will reflect all messages handled at every node, therefore resembling overhead for the entire network. Please note that the scenario of starting all nodes from scratch is quite extreme and it serves the purpose of comparing Babel to Babel-SFE.

Fig. 4 depicts the time until routing convergence is achieved in the whole network topology of 50 up to 300 nodes. We assume Babel-SFE to be more efficient and faster as the original Babel by design. Overall both approaches deliver fast convergence times under 30s on initial network boot. Babel-SFE is always faster or as least as fast as Babel with 50 nodes. These results indicate reliable transport of routing messages

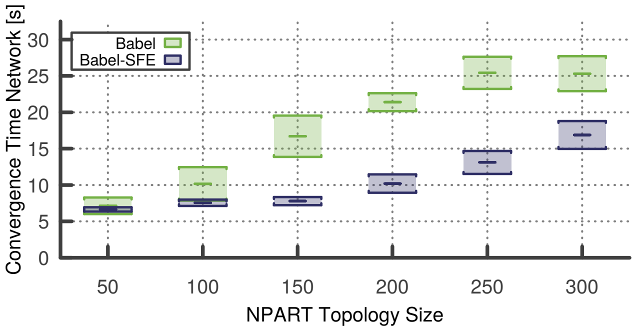

Fig. 4. Time until routing convergence is achieved.

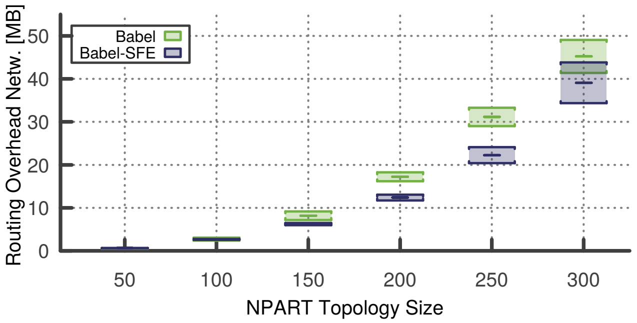

Fig. 5. Network overhead until routing convergence is achieved.

to be advantageous concerning convergence time, even tough they introduce a slight overhead.

Moreover, Fig. 5 depicts the network overhead of both Babel and Babel-SFE on network boot. We first expected Babel-SFE to be significantly better than Babel in this scenario. But the data only shows a small difference. Therefore we can conclude that the initial boot process has to spread topology knowledge in the whole network and does only profit by a small margin from our new mechanisms.

A completely different view on overhead can be obtained, when the experiment timespan is increased, see Fig. 6. After booting the network and an initially higher overhead, our mechanisms become effective – reducing the overhead close to zero when using Babel-SFE. With regular Babel the overhead in large networks increases with every update interval, as Babel has to retransmit every update.

V. CONCLUSION AND FUTURE WORK

In this article, we presented the mechanism of Spare Forwarding Entries (SFEs), an extension to regular distance vector routing providing significantly more robustness against outages in an efficient manner (with respect to data rate and airtime). Integrating SFEs into the well-known Babel routing protocol enabled simulation studies, which indicate a much better PDR in outage scenarios. Additional modifications to Babel leave periodic updates behind, consequently, relying on reliable transport of routing messages. We were able to reduce routing overhead slightly and initial convergence times significantly.

This work is only one of many building blocks needed for large and scalable, distributed mesh networks. Future work is necessary in automatic multi-channel selection strategies along with distributed client roaming assistance, e.g., estimated neighbor lists for accelerated scans from wireless clients.

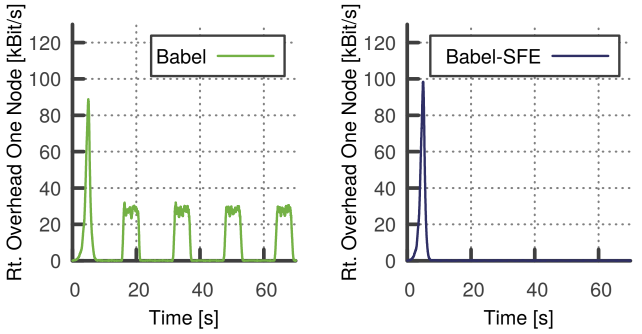

Fig. 6. Routing overhead for one node (NPART topology with 100 nodes) over the course of 70 s (average of 32 runs).

Regarding Babel-SFE in particular, more outage scenarios have to be considered in the future.

REFERENCES

[1] S. Sampaio, P. Souto, and F. Vasques, “A Review of Scalability and Topological Stability Issues in IEEE802.11s Wireless Mesh Networks Deployments,” *Intl. Journal of Communication Systems*, 2016.

[2] T. Clausen and P. Jacquet, “Optimized Link State Routing Protocol (OLSR),” RFC 3626, 2003.

[3] D. Johnson, N. Ntlatlapa, and C. Aichele, “Simple Pragmatic Approach to Mesh Routing Using BATMAN,” in *IFIP Intl. Symposium on Wireless Communications and Inf. Techn. in Developing Countries*, 2008.

[4] D. Murray, M. Dixon, and T. Koziniec, “An Experimental Comparison of Routing Protocols in Multi Hop Ad Hoc Networks,” in *Australasian Telecommunication Networks and Applications Conference*, 2010.

[5] J. Chroboczek, “The Babel Routing Protocol,” RFC 6126, 2011.

[6] V. Chaudhary, U. Patel, Shivaji, and R. Kumar, “AODV-MBR: A New Routing Protocol with Modified Backup Approach,” in *Communications in Computer and Information Science*. Springer, 2012.

[7] BMX6 Mesh Networking Protocol. [Online]. Available: https://bmx6.net

[8] A. Neumann, L. Navarro, and L. Cerdà-Alabern, “Enabling Individually Entrusted Routing Security for Open and Decentralized Community Networks,” *Ad Hoc Networks*, 2018.

[9] D. Savage, J. Ng, S. Moore, D. Slice, P. Paluch, and R. White, “Cisco’s Enhanced Interior Gateway Routing Protocol (EIGRP),” RFC 7868, 2016.

[10] J. J. Garcia-Lunes-Aceves, “Loop-Free Routing Using Diffusing Computations,” *IEEE/ACM Transactions on Networking*, 1993.

[11] S. Vutukury and J. Garcia-Luna-Aceves, “A Simple Approximation to Minimum-Delay Routing,” in *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 1999.

[12] S. Ray, R. Guérin, and R. Sofia, “Distributed Path Computation Without Transient Loops: An Intermediate Variables Approach,” in *Intl. Teletraffic Congress*. Springer, 2007.

[13] S. Vutukury and J. J. Garcia-Luna-Aceves, “MDVA: A Distance-Vector Multipath Routing Protocol,” in *IEEE Conference on Computer Communications*, 2001.

[14] S. Vutukury and J. J. Garcia-Luna-Aceves, “An Algorithm for Multipath Computation Using Distance-Vectors with Predecessor Information,” in *Intl. Conference on Computer Communications and Networks*, 1999.

[15] R. Jansen, S. Hanemann, and B. Freisleben, “Proactive Distance-Vector Multipath Routing for Wireless Ad Hoc Networks,” in *Symposium on Communications and Vehicular Technology*, 2003.

[16] M. Backhaus, M. Theil, M. Rossberg, and G. Schaefer, “Robust and Scalable Routing in Wireless Mesh Networks Using Interference-Disjoint Backup Paths,” in *IFIP Wireless and Mobile Netw. Conference*, 2019.

[17] The Babel Routing Daemon Git Repository. Accessed: 20-Jan-2020. [Online]. Available: https://github.com/jech/babeld

[18] M. Backhaus, M. Theil, M. Rossberg, G. Schaefer, and D. Sukiennik, “A Comprehensive Framework to Evaluate Wireless Networks in Simulation and Real Systems,” in *Intl. Symposium on Distributed Simulation and Real Time Applications*, 2018.

[19] G. F. Riley and T. R. Henderson, *The ns-3 Network Simulator*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[20] B. Milic and M. Malek, “NPART - Node Placement Algorithm for Realistic Topologies in Wireless Multihop Network Simulation,” in *Simutools*, 2009.