Efficiency of BIER Multicast in Large Networks

Daniel Merling*, Thomas Stüber*, Michael Menth

Chair of Communication Networks, University of Tuebingen, Germany
{daniel.merling, thomas.stueber, menth}@uni-tuebingen.de

***Abstract*—Bit Index Explicit Replication (BIER) has been introduced by the IETF to transport IP multicast (IPMC) traffic within a BIER domain. Its advantage over IPMC is improved scalability regarding the number of multicast groups. However, scaling BIER to large networks is a challenge. To that end, receivers of a BIER domain are assigned to smaller subdomains. To deliver an IPMC packet over a BIER domain, a copy is sent to any subdomain with a receiver for that packet. Consequently, some links may carry multiple copies of the same IPMC packet, which contradicts the multicast idea.**

In this paper, we propose and compare various algorithms to select subdomains for BIER in order to keep the overall BIER traffic low despite multiple packet copies. We apply them to investigate the traffic savings potential of IPMC and BIER relative to unicast under various conditions. We show that the traffic savings depend on network topology, network size, and the size of the multicast groups. Also the extra traffic caused by BIER depends on these factors. In spite of some redundant packets, BIER can efficiently reduce the overall traffic in most network topologies. Similarly to IPMC, BIER also avoids heavily loaded links. Finally, we demonstrate that BIER subdomains optimized for failure-free conditions do not cause extensive overload in case of single link failures.

***Index Terms*—Bit Index Explicit Replication (BIER), multicast, IP networks, performance evaluation, optimization**

I. Introduction

IP multicast (IPMC) reduces the traffic load of one-to-many traffic [1], e.g., Multicast VPN, streaming, content delivery networks, or data center virtualization/overlay because it avoids redundant packet copies. To that end, it distributes traffic of a multicast group along a tree so that any link in an IP network forwards at most a single copy of a packet. However, all core nodes that are part of a distribution tree of an IPMC group need to maintain forwarding state for that IPMC group. This approach causes a threefold scalability issue. First, core nodes need to maintain possibly extensive forwarding information bases (FIBs). Second, when subscriber change, the core nodes of the affected IPMC group require updates which cause serious signaling efforts. Third, if links or nodes fail, or the topology changes, the traffic of many multicast groups may be affected so that many routers experience a large signaling load. The IETF has proposed Bit Index Explicit Replication (BIER) [2] to counteract that problem. BIER tunnels multicast traffic through a BIER domain and delivers a copy to each desired egress node. BIER solves the scalability problem by keeping the core nodes of the BIER domain unaware of any multicast group. Nevertheless, scaling BIER to large networks is a challenge. Multiple copies of a multicast packet may need to be forwarded over the same link, which contracts the multicast idea and may prevent BIER from efficiently reducing the traffic load for multicast traffic. We briefly explain the reason and provide the ground for this research work.

When an ingress node of a BIER domain receives an IPMC packet, it adds a BIER header including a bitstring. The positions in the bitstring correspond to egress nodes of the BIER domain and the activated bits indicate the receivers of the BIER packet. The bitstring enables BIER routers to forward BIER packets without knowing multicast groups. As the bitstring has a limited size, BIER domains with more egress nodes require a scaling feature. Subdomains are introduced which are sets of egress nodes, and bitstrings are defined for each subdomain. Thus, if an IPMC packet needs to be forwarded to egress nodes in different subdomains, multiple BIER packets with different bitstrings are sent and possibly pass identical links. This obviously reduces the efficiency of BIER to distribute multicast traffic compared to IPMC. Thus, BIER enables stateless transport of multicast traffic and thereby mitigates IPMC's scalability problem. However, it is less efficient than IPMC with regard to traffic load reduction.

The contributions of this paper are manifold. First, we show that a simple application of BIER's scaling feature [2], i.e., random subdomain clustering, cannot efficiently reduce traffic load in the network . Second, we present means to compute efficient subdomain clusterings. To that end, we describe an integer-linear program (ILP) that computes subdomain clustering in a way that minimizes the overall traffic load in the network. We also design a heuristic to approximate the solution of the ILP because it works only on small topologies. Third, we quantify and compare the ability of IPMC and BIER to efficiently reduce the load from multicast traffic in comparison to unicast. In particular, we evaluate the efficiency of BIER with the proposed subdomain clustering mechanisms and compare it to a naive application of BIER's scaling feature. We define suitable metrics and show that the efficiency of multicast depends on network topology and size as well as the size of the multicast groups. Fourth, we investigate the effect of link failures on the efficiency of BIER with optimized subdomains. This is interesting as link failures change the routing based on which the subdomains were optimized.

The remainder of the paper is structured as follows. In the next section we review related work. Section III gives a primer

The authors acknowledge the funding by the Deutsche Forschungsgemeinschaft (DFG) under grant ME2727/1-2. The authors alone are responsible for the content of the paper.

*These authors contributed equally

Published in IEEE Transactions on Network and Service Management, vol. 20, no. 4, pp. 4013-4027, December 2023.

on BIER and shows that BIER generates a separate packet copy for almost every subdomain even for small multicast groups. In Section IV we propose algorithms to compute subdomains for BIER networks. We compare the algorithms with regard to runtime and quality in Section V. Section VI evaluates and compares the traffic savings potential of IPMC and BIER for multicast traffic. In Section VII we evaluate the efficiency of BIER in case of single link failures. Finally, we conclude the paper in Section VIII.

II. Related Work

We review advances for IPMC and BIER-based multicast and mention well-known clustering algorithms.

A. Advances for IPMC

Islam et al. [3] and Al-Saeed et al. [4] provide comprehensive surveys for multicast. Most of the cited papers discuss shortcomings of IPMC as already mentioned in the introduction, i.e., limited scalability in terms of signaling and state overhead. Many approaches aim to make traditional IPMC forwarding more efficient. Intelligent mechanisms for multicast tree building are presented to reduce the size of the forwarding information base (FIB), or efficient signaling mechanisms are proposed. However, they counteract the shortcomings of traditional IPMC only up to the point where the inherent design flaw of traditional IPMC, i.e., maintaining IPMC-group-dependent state in core devices, causes significant overhead, and therefore scalability issues.

Elmo [5] improves the scalability of traditional IPMC in data centers. Multicast group information is encoded in packet headers to reduce the FIB of core nodes by leveraging characteristic properties of data center topologies. The Avalanche Routing Algorithm (AvRA) [6] also leverages properties of data center networks to optimize link utilization of distribution trees. Dual-Structure Multicast (DuSM) [7] builds specialized forwarding structures for high-bandwidth and low-bandwidth flows. It improves scalability and link utilization in data centers.

Zhang et al. [8] optimize application layer multicast (ALM). They continuously monitor the application-specific distribution tree and update its structure according to the optimization objective of the multicast group. The authors of [9] study the distribution of delay-sensitive data with minimum latency. They propose a set of algorithms that construct minimum-delay trees for different kinds of application requirements like min-average, min-maximum, real-time requirements, etc. Li et al. [10] leverage the structure of data center networks to improve the scalability of traditional multicast. They optimize the forwarding tables by partitioning the multicast address space and aggregating multicast addresses at bottleneck switches. Kaafar et al. [11] present a new overlay multicast tree construction scheme. It leverages location-information of subscribers to build efficient distribution trees.

Software-Defined Multicast (SDM) [12] is a well-managed multicast platform. It is specialized on P2P-based video streaming for over-the-top and overlay-based live streaming services. In [13] traffic engineering features are added to SDM. Lin et al. [14] propose to share distribution trees between multicast groups to reduce the size of the FIB in core nodes and implement it in OpenFlow. Similarly, the authors of [15] leverage bloom filters to reduce the number of TCAM-entries in software-defined networks. Adaptive SDN-based SVC multicast (ASCast) [16] optimizes multicast forwarding for video live streaming by minimizing latency and delay. To that end, the authors propose an integer linear program for optimal tree building, and TCAM-based forwarding tables for fast packet processing. Humernbrum et al. [17] reduce the size of the FIB in some core nodes by introducing address translation from multicast addresses to unicast addresses at the last multicast hop. Jia et al. [18] reduce the size of the FIB in core nodes and facilitate efficient implementations. They leverage prime numbers and the Chinese remainder theorem to efficiently organize FIB structures. Steiner trees [19] are well-researched structures to build efficient multicast trees. Many papers modify and extend Steiner trees to build specialized multicast trees that minimize specific aspects like link costs [20], number of branch nodes [21], number of hops [22], delay [23], optimal placement of IPMC sources [24], or retransmission efficiency [25].

B. Advances for BIER

BIER uses a novel header and its forwarding behavior distinguishes substantially from IP forwarding. That is, BIER does not require per-IPMC-group-state in its core devices. Therefore, it does not suffer the same scalability issues as IPMC. Giorgetti et al. [26], [27] show a first implementation of BIER in OpenFlow. Merling et al. [28] present a BIER prototype for a P4-programmable software switch with a throughput of around 900 Mb/s. In a follow-up work [29] they implement BIER for the P4-programmable switching ASIC Tofino that supports 100 Gb/s throughput per port. They also propose how BIER traffic should be rerouted in case of failures, which has been adopted as IETF working group document [30].

The authors of [31] evaluate the retransmission efficiency of BIER when subscribers signal missing packets by negative acknowledgments, i.e., NACKs. Traditional IPMC leverages either unicast packets or retransmission to the entire multicast group when some subscribers signal NACKs. The BIER header allows to retransmit packets to specific subscribers only, i.e., NACK senders, while sending only one packet copy over each link. The authors find that BIER causes less overhead in terms of number of retransmitted packets and that it achieves better link utilization. Desmouceaux et al. [32] increase efficiency of retransmission with BIER by allowing intermediate nodes to resend packets, if possible, instead of resending the packet at the source. This significantly reduces the overall retransmission traffic.

Eckert et al. [33] propose tree engineering for BIER, i.e., BIER-TE. It leverages the BIER header to also encode the distribution tree of a packet in terms of traversed links. In [34] 1+1 protection for BIER is presented using maximally

redundant trees (MRTs). Traffic is distributed simultaneously over two disjoint trees so that packets are delivered even if one tree is compromised by a failure.

C. Clustering Algorithms

In this work we cluster receivers of BIER domains into subdomains. Karypis et. al. [35] present an algorithm to compute a bisection of a graph by performing a breadth-first search starting from two center nodes. The authors of [36] propose a similar method to compute k-partitions for arbitrary k, using k center nodes. Instead of two breadth-first searches, this algorithm performs k breadth-first searches in parallel. The resulting partitions tend to reduce the number of border nodes instead of cross-edges, which is a good property for load balancing. The approach is closely related to k-means clustering with Lloyd's algorithm [37]. The algorithm selects k center nodes and adds all nodes to the cluster with the nearest center node. The center nodes are readjusted to reflect the center of the clusters and this step is repeated until no changes occur. k-means clustering is not suitable for our problem, as cluster sizes cannot be limited. In contrast to that, the bubble-growing approach of [36] produces equal size partitions. The heuristic algorithm for BIER clustering in this work follows a similar approach.

III. Bit Index Explicit Replication (BIER)

In this section we introduce fundamentals of BIER and explain its scaling mechanism for large networks. In addition, we show that the mechanism tends to produce multiple BIER packets for a single IPMC packet, even for small multicast groups.

A. Overview

BIER is a domain-based mechanism to transport IPMC traffic over a so-called routing underlay network, e.g., an IP network [2]. Figure 1 shows the layered BIER architecture.

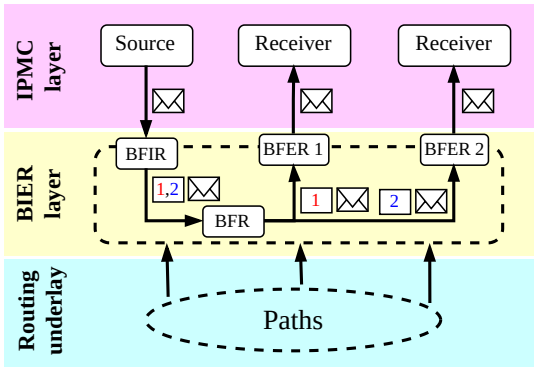


Fig. 1: Layered BIER architecture according to [28].

BIER-capable routers are called bit forwarding routers (BFRs). Ingress and egress nodes of a BIER domain are called bit forwarding ingress and egress routers (BFIRs, BFERs). The BIER header contains a bitstring with bit positions for all BFERs. BFIRs encapsulate IPMC traffic with a BIER header and the activated bits in its bitstring indicate the set of BFERs that are connected to subscribed IPMC clients, and hence, should receive a copy of the packet. BFRs forward BIER packets based on this bitstring along a tree towards the indicated BFERs. Thereby, only a single copy is sent over each involved link. The paths of the tree are inherited from the routing underlay but BIER-encapsulated IPMC packets are usually sent over Layer 2 technology. BFERs remove the BIER header from the packets and pass them to the IPMC layer.

B. Scaling BIER to Large Networks

BIER hardware must implement a bitstring length of 256 bits, but larger bitstrings, e.g., 1024 bits, may also be supported [2]. However, large bitstrings increase the header size, which is tolerable only to some extent. Any BFER requires a position in the bitstring to be addressable. To make BIER applicable to networks with more BFERs than the size of the bitstring, so-called BIER subdomains are introduced. BIER subdomains are identified by their subdomain identifier (SDI) and they define different mappings of BFERs to bit positions for the subdomain-specific bitstring in the BIER packet header. Therefore, only the combination of SDI and bitstring in the BIER packet header determines the addressed BFERs of that packet. If a BFIR receives an IPMC packet, it sends a packet copy of that IPMC packet to each subdomain that contains at least one receiver, i.e., BFER. Thereby, the BFIR encapsulates the packet copies with a BIER header with the right SDI and bitstring to address the subscribers in each subdomain.

C. BIER Packets Needed for Single IPMC Packet

When a BIER domain is large, it may require multiple subdomains. Then, the BFERs of a BIER domain are assigned to bit positions in the bitstrings of different subdomains. As a consequence, when an IPMC packet is to be carried through a BIER domain, multiple BIER packets with different SDIs may be created to address all desired receivers. We call them redundant packet copies as they carry the same IPMC packet. They cause extra traffic and reduce BIER's ability to reduce load from multicast traffic compared to normal IPMC forwarding.

We investigate how many different BIER packets are generated on average when a BFIR sends an IPMC packet over a BIER domain. To that end, we consider a BIER domain with $n = 1024$ BFERs and bitstring lengths of $b \in \{128, 256, 512, 1024\}$ bits. Hence, $s \in \{1, 2, 4, 8\}$ subdomains are needed to provide all BFERs with bit positions. We use a Markov chain model to compute the average number of different BIER packets needed if an IPMC packet has r BFERs as receivers; thereby we assume that receivers of a packet belong with equal probability to any of the subdomains.

Figure 2 shows that the average number of BIER packets significantly depends on the number of receivers r and the number of subdomains s. The number of BIER packets converges quickly to the number of subdomains s. If $r = 3 \cdot s$ receivers are addressed, almost s different BIER packets need to be sent for a single IPMC packet.

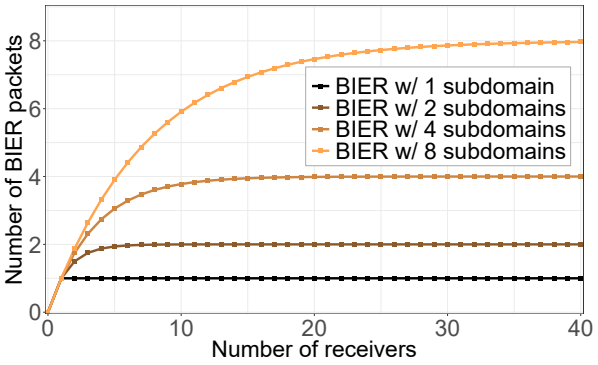


Fig. 2: Average number of redundant BIER packet copies needed to forward a single IPMC packet through a BIER domain with $n = 1024$ BFERs partitioned into $s \in \{1, 2, 4, 8\}$ subdomains.

As the number of redundant packets is large even for small multicast groups, it is relevant to study their impact on the overall extra traffic in the network and on the ability of BIER to efficiently carry multicast traffic compared to IPMC. Moreover, the effect of redundant BIER packets may be mitigated. If a specific part of the BIER domain accommodates only BFERs from a single subdomain, only packets for that subdomain will be forwarded to that part of the network, which avoids redundant packets in this area. Thus, when subdomains are chosen appropriately, BIER may be able to deliver multicast traffic with only little extra traffic compared to IPMC.

IV. Algorithms for BIER Clustering

As explained in the previous section, sudomains for BIER domains should be defined such that the overall load from multicast traffic is low in the entire network even if multiple redundant BIER packets need to be sent to BFERs in different subdomains.

We first formalize this challenge as the "BIER clustering problem". Then, we propose three classes of algorithms to assign BFERs to subdomains of a BIER domain: random, optimal, and heuristic. For optimal solutions, we propose topology-specific algorithms for selected, regular topologies. For arbitrary topologies we propose an integer linear program (ILP) to optimally solve the BIER clustering problem. Finally, we suggest a heuristic algorithm that may be used when the ILP is not solvable for complexity reasons.

A. The BIER Clustering Problem

We introduce nomenclature and constraints as well as the objective function for BIER clustering, and discuss alternate optimization goals. Finally, we discuss the investigated topology-types.

1) Nomenclature and Constraints: A network topology is given by a set of n vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$. The set of edges on the path between any two nodes $v, w \in \mathcal{V}$ is denoted by $p(v, w) \subseteq \mathcal{E}$; it is inherited by the routing underlay. The objective of the clustering is to find a set of subdomains $\mathcal{C}$ so that any subdomain $\mathcal{S} \in \mathcal{C}$ is a subset of all nodes $\mathcal{S} \subseteq \mathcal{V}$ and the union of all subsets covers all nodes, i.e., $\bigcup_{\mathcal{S}\in\mathcal{C}} \mathcal{S} = \mathcal{V}$. Moreover, the size of a subdomain is limited by the length of the bitsring b.

In theory, there is no limit on the number of subdomains and subdomains may overlap. However, more subdomains imply more forwarding information on the BFRs, more complex bitstring definition for a multicast group, and longer subdomain identifiers. Therefore, we keep the number of subdomains as low as possible. We further assume that any node of the BIER domain is a BFER. Therefore, the number of subdomains is $s = \lceil \frac{n}{b} \rceil$. We require the subdomains to be disjoint. This simplifies the algorithms and has no impact on the results as the network sizes in our experiments are multiples of maximum subdomain sizes.

2) Objective Function: The objective function for the clustering is to keep the overall traffic low. We define the overall traffic as the number of packets carried over all links of the BIER domain when every node sends a packet to every other node. The traffic induced by a BIER packet sent from a single BFIR v to all BFERs within a subdomain $\mathcal{S}$ is the number of edges traversed, i.e., $\left|\bigcup_{w\in\mathcal{S}} p(v, w)\right|$. Thus, the overall BIER traffic load is

$$\rho = \sum_{v\in\mathcal{V}} \sum_{\mathcal{S}\in\mathcal{C}} \left| \bigcup_{w\in\mathcal{S}} p(v, w). \right| \tag{1}$$

This metric is to be minimized by a clustering $\mathcal{C}$.

3) Alternate Optimization Goals: Future work may optimize the clustering so that more subdomains are allowed. Then, some subdomains may overlap and BFIRs can choose over which subdomains a BFER will be reached. This increases the potential for the minimization of overall traffic so that even fewer redundant BIER packets may need to be generated. This, however, requires the knowledge of the IPMC groups and needs more resources on BFRs and BFIRs.

Another optimization approach is monitoring BIER traffic and building subdomains according to higher-level information, e.g., multicast groups, traffic patterns, proximity, etc. However, such methods are significantly more complex than the suggested approach, require additional research, and are out of scope of this paper.

4) Studied Topologies: We include two types of topologies in our study. First, we leverage full mesh, line, ring, and perfect binary tree to reveal interesting corner cases for multicast traffic. That is, multicast cannot reduce traffic in full-mesh topologies but has its greatest effect in line and ring topologies. Perfect-binary trees have a simple but non-random and non-trivial topology structure which is why we included them in the evaluations. The evaluations of such best/worst cases gives insights in upper and lower bounds. Second, we conduct experiments on sets of random mesh topologies with different average node degrees to analyze the performance of the studied mechanisms on more realistic topologies.

B. Random BIER Clustering

We briefly explain random BIER clustering. A bitstring length of b is given. A set of n BFERs is subdivided into equal-size $s = \lceil \frac{n}{b} \rceil$ subdomains. BFERs are randomly assigned to these subdomains whereby their size is limited to b BFERs. In Section V-C we use this algorithm as a baseline for comparison.

C. Optimal BIER Clustering for Selected Topologies

We describe optimal clusterings for selected, regular topologies: full mesh, line, ring, and perfect binary tree. We renounce on a formal proof of optimality as this is rather obvious.

1) Full Mesh: Here, random assignment is optimal. In full meshes, all traffic is exchanged over a direct link between source and destination because all nodes are neighbors. However, in such topologies, there is no traffic reduction potential for multicast and we do not consider full meshes any further.

2) Line Topologies: Start at one end of the line. Assign the next b neighboring nodes to a subdomain. Repeat until all nodes are assigned. The last subdomain may have less than b nodes.

3) Ring Topologies: Select an arbitrary position in the ring and choose a direction. Assign the next b neighboring nodes to a subdomain. Repeat until all nodes are assigned. The last subdomain may have less than b nodes.

4) Perfect Binary Trees: We consider a perfect binary tree. The depth of a node is its distance to the root plus one so that the leaves have maximum depth. We denote their depth as the height h of the tree. We state that a perfect binary tree with height h has $2^h - 1$ nodes.

We assume that the bitstring size is $b = 2^k$. It can accommodate a perfect binary tree with height k. We give an algorithm to cluster a perfect binary tree with height h into 2^{h-k} subdomains with up to 2^k nodes. We take all subtrees with roots of depth $h-k+1$ as initial subdomains. The other unassigned nodes are assigned to a nearest possible subdomain which still accepts additional nodes. Thereby, the assignment order of these nodes is inverse to their depth. The order among nodes with equal depth does not matter.

D. Optimal BIER Clustering for Arbitrary Topologies

We first explain fundamentals of integer linear programs (ILPs). Then, we apply them for optimal clustering of BIER domains.

1) Fundamentals of ILPs: An ILP describes the solution space of an optimization problem with so-called decision variables and linear inequalities. Parameters of the optimization problem serve as coefficients in the inequalities. A linear objective function describes the quality of possible solutions and is to be minimized.

ILP solvers find the best integer solution for decision variables that fulfill all inequalities. During the solution process, an ILP solver indicates lower and upper bounds regarding the objective value for the best solution. The upper bound is the value for the best solution found so far. While progressing, better solutions may be found and the lower bound for the best solution may increase. If upper and lower bound meet, the ILP solver found an optimal solution.

2) BIER Clustering Using ILPs: We build an ILP that describes the solution space for BIER clustering and an objective function for the overall traffic load given in Equation (1). Its output is an optimal clustering $\mathcal{C}$ of the network that minimizes the objective function.

$$\forall v \in \mathcal{V}: \quad \sum_{\mathcal{S} \in \mathcal{C}} x_v^{\mathcal{S}} = 1 \tag{2}$$

$$\mathcal{S} \in \mathcal{C}: \quad \sum_{v \in \mathcal{V}} x_v^{\mathcal{S}} \leq b \tag{3}$$

$$\forall v, w \in \mathcal{V}, e \in \mathcal{E}, \mathcal{S} \in \mathcal{C}: \quad p_{e,v,w} \cdot x_w^{\mathcal{S}} \leq y_{v,e}^{\mathcal{S}} \tag{4}$$

$$\forall v \in \mathcal{V}, e \in \mathcal{E}, \mathcal{S} \in \mathcal{C}: \quad y_{v,e}^{\mathcal{S}} \leq \sum_{w \in \mathcal{V}} p_{e,v,w} \cdot x_w^{\mathcal{S}} \tag{5}$$

$$\text{min: } \rho = \sum_{v \in \mathcal{V}} \sum_{\mathcal{S} \in \mathcal{C}} \sum_{e \in \mathcal{E}} y_{v,e}^{\mathcal{S}} \tag{6}$$

The ILP is given by Equation (2), Inequalities (3)–(5), and the objective function (6). It contains two types of binary decision variables. The decision variable $x_v^{\mathcal{S}}$ indicates whether node v belongs to subdomain $\mathcal{S}$; it is 1 if $v \in \mathcal{V}$ is in subdomain $\mathcal{S}$, otherwise it is 0. Equation 2 enforces that any node is part of exactly one subdomain. Inequalitiy 3 ensures that a subdomain contains at most b nodes. The decision variable $y_{v,e}^{\mathcal{S}}$ indicates whether edge e is part of the multicast tree from node v to any node $w \in \mathcal{S}$. It depends on $x_v^{\mathcal{S}}$ and the forwarding information. The latter is given by coefficients $p_{e,v,w}$ which are 1 if edge e is on the path from v to w; otherwise the coefficient is 0. This dependency is modelled by Inequalities 4 and 5. Equation 4 ensures that $y_{v,e}^{\mathcal{S}} = 1$ if e is part of the path from BFIR v to any BFER w in subdomain $\mathcal{S}$. Equation 5 ensures that the decision variable $y_{v,e}^{\mathcal{S}}$ is 0 if e is not part of any path from v (BFIR) to any w (BFER) in $\mathcal{S}$; thereby the membership $w \in \mathcal{S}$ is expressed only indirectly by $w \in \mathcal{V}$ and the decision variable $x_w^{\mathcal{S}}$.

The objective function in Equation (6) quantifies the overall traffic as defined in Equation (1) and is to be minimized.

E. Heuristic BIER Clustering

We propose a heuristic clustering algorithm that consists of two phases. Phase 1 selects initial subdomains. Phase 2 improves these subdomains according to Equation (1) by exchanging the assignment of node pairs to their subdomains.

Phase 1 works as follows. First, randomly select s nodes as center nodes of the different subdomains. Second, add further nodes to the subdomains until their maximum size b is reached. To that end, nearest non-assigned nodes are assigned to the center nodes in round-robin fashion. This yields a clustering of the BIER domain into subdomains. We repeat Phase 1 to generate $10 \cdot s$[†] clusterings and choose the best according to

[†]We performed evaluations with significantly higher repetitions but observed no increase in quality. Thus, we selected 10 runs as a reasonable basis to find a good solution.

the objective function in Equation (1) to continue with it in Phase 2.

Phase 2 improves the clustering. First, randomly select two nodes that have neighbors in other subdomains and that are assigned to different subdomains. Swap their assignment if this reduces the overall load according to Equation (1). Repeat this procedure until ρ from Equation (1) does not decrease for $n = |\mathcal{V}|$ steps. When computing a clustering for a network, we perform the presented algorithm 20 times and take the best solution.

This algorithm is simple but works better than more complex approaches we have evaluated before. We evaluate the quality of this heuristic in the next section.

V. Comparison of BIER Clustering Algorithms

In this section we compare the BIER clustering algorithms from the previous section with regard to runtime and quality. First we present the topologies that we use for evaluations in this paper. Then, we demonstrate that the runtime of the ILP-based optimization is feasible only for small networks. Finally, we compare the quality of the subdomains obtained for different algorithms, topologies, and network sizes.

A. Topologies

In this work we investigate delivery of multicast traffic in various network topologies: full mesh, line, ring, perfect binary tree, and mesh networks with node degree $d \in \{2, 4, 6, 8\}$. We refer to the latter as mesh-d. We construct them using the topology generator BRITE [38] which leverages a Waxman model [39]. While the first mentioned topologies are regular so that there is only a single choice for a network with n nodes, mesh-d networks are randomly constructed. Therefore, we generate 10 different representatives and compute average values for the considered metrics. The 95% confidence intervals are below 0.3% for all reported results so that we omit them in all tables and figures.

Topology	$n = 64$		$n = 128$	
	$s = 2$	$s = 4$	$s = 2$	$s = 4$
Line	0.11	3.80	1.07	45.51
Ring	66.51	21139.70	3633.59	-
Perfect binary tree	0.11	1.10	0.33	6.71
Mesh-2	0.06	3.59	0.21	22.67
Mesh-4	76.09	-	-	-
Mesh-6	718.23	-	-	-
Mesh-8	3883.62	-	-	-

Tab. 1: Time to solve ILPs for BIER clustering in seconds. Some instances could not be solved within 72 hours.

B. Runtime for ILP-Based Optimization

We measure the runtime to solve ILPs for BIER clustering with the ILP solver Gurobi 9.1 on a Ryzen 3900X CPU with 12 cores running at 3.8 GHz with 64 GB RAM.

Table 1 compiles the runtimes of the solver for different network topologies, network sizes, and number of subdomains. Perfect binary trees have one node less than indicated in the table. The runtime to solve the ILPs increases with network size and in particular with the number of subdomains. The network topology also has a significant impact. For some topologies, networks with 128 nodes or with 4 subdomains cannot be solved within three days.

In contrast, the heuristic algorithm has a runtime of a few seconds for any topology with $n = 1024$ nodes, and $s = 4$ subdomains. For the largest networks with $n = 8192$ nodes and $s = 32$ subdomains, it takes 8–16 h for mesh-4 and mesh-6, and 16–24 h for lines, perfect binary trees, mesh-2, and mesh-8. Only very large rings with $n = 8192$ nodes required around 3 days.

Thus, solving the ILP for optimal BIER clustering is infeasible for realistic problem instances, but the runtime of the heuristic algorithm is acceptable even for large networks. Therefore, we utilize for the evaluations in Section VI the topology-specific solutions of Section IV-C for lines, rings, and perfect binary trees, and the heuristic algorithm for mesh-d networks.

C. Quality Comparison

We now compare the quality of heuristic results with those from optimal and random subdomain assignment. The metric is the overall traffic load ρ with BIER when every node sends a packet to any other node (see Equation (1)).

We first consider mesh-d, for which only the ILP-based algorithm can deliver optimal results but only for small networks. Table 2 shows the overall traffic for subdomains generated with heuristic and with random assignment relative to the overall traffic for optimal subdomains. All heuristic results are close to optimum. We observe for mesh-2 that larger networks and more subdomains slightly degrade the results of the heuristic algorithm. Random assignment is clearly worse, i.e., it generates 33%-80% more extra traffic than optimal subdomains while heuristic assignment causes only 0.3%-1.5% more extra traffic. The quality of the heuristic results tends to improve with increasing node degree.

Topology	n	s	Heuristic (%)	Random (%)
Mesh-2	64	2	100.3	132.6
		4	100.7	162.2
	128	2	100.5	133.7
		4	101.5	179.8
Mesh-4	64	2	100.3	115.2
Mesh-6	64	2	100.4	110.6
Mesh-8	64	2	100.3	107.1

Tab. 2: Overall traffic load for heuristic and random BIER clustering depending on network size n and number of subdomains s; numbers are relative to the overall traffic load for optimal subdomains computed based on ILP solutions.

Now we discuss larger, regular topologies for which the algorithms of Section IV-C provide optimal results. We cluster the networks into subdomains of size $b = 256$. The results are compiled in Table 3. We consider networks with $n \in \{256, 512, 1024, 2048, 4096, 8192\}$ nodes, an exception are perfect binary trees with only $n - 1$ nodes. Consequently, multiple subdomains $s \in \{1, 2, 4, 8, 16, 32\}$ are needed. The

overall traffic load is given relative to the one for optimal subdomains.

n	s	Line (%)		Ring (%)		Perfect binary tree (%)	
		Heur.	Rnd.	Heur.	Rnd.	Heur.	Rnd.
256	1	100	100	100	100	100	100
512	2	100	159.5	100	133.1	101.2	142.8
1024	4	100	212.2	100	199.1	100.8	197.2
2048	8	100	249.3	100	265.1	100.6	262.5
4096	16	100	271.8	108.7	317.9	104.9	336.9
8192	32	100	284.3	134.1	353.0	118.0	416.9

Tab. 3: Overall traffic load for heuristic and random BIER clustering depending on network size n and number of subdomains s; numbers are relative to the overall traffic load for optimal subdomains computed based on topology-specific solutions.

We observe that the quality of the heuristic is almost optimal for up to 2048 nodes. Beyond that, the quality degrades by up to 34% for rings compared to optimum. The quality for lines and perfect binary trees is better with a degradation of at most 18%. The results with random assignment are much worse than those with optimum and heuristic assignment.

We draw two major conclusions. First, optimization of subdomains is important as random subdomains are likely to cause a lot more extra traffic than needed in large BIER domains. Second, subdomains obtained through the presented heuristic are almost optimal for networks up to 2048 nodes, beyond that we see a degradation. However, even then heuristic subdomains are still much better than random subdomains. The heuristic is needed for the evaluation of mesh-d networks in Section VI. We believe that the quality of the heuristic results for mesh-d is acceptable even for large networks because the heuristic algorithm performed well in large networks for lines, rings, and perfect binary trees. Therefore, the method may be suitable for application in practice.

VI. Traffic Savings with IPMC and BIER

In this section we investigate the potential of multicast variants, i.e., IPMC and BIER, to reduce the traffic load from multicast traffic relative to unicast, and compare it with each other. We first discuss the methodology. Afterwards we study the reduction potential for overall traffic depending on network size and multicast group size. Then, we show that both IPMC and BIER can well avoid heavily loaded links. Finally, we examine the impact of header size on the traffic saving potential of BIER.

A. Methodology

We describe the general evaluation approach, investigated network topologies, the way BIER subdomains are clustered in the study, packet sizes, evaluation metrics, and identified influencing factors.

1) General Approach: It is obvious that multicast groups can be very different, both in size and geographical distribution. Moreover, networks supporting multicast can have different topology. As those factors likely impact the efficiency of multicast variants, we study them depending on network topology, network size, and multicast group size. We study the topologies presented in Section V-A; if the topologies are random, we report averages from 10 different topologies and omit the small confidence intervals as mentioned. The networks have $n \in \{256, 512, 1024, 2048, 4096, 8192\}$ nodes, with the exception of perfect binary trees that have only network size $n - 1$.

In this section we evaluate the traffic saving potentials of IPMC and BIER in comparison to unicast and to each other. We simulate the transmission of a single packet from every source to all subscribers of a multicast group. We describe the models for multicast groups in the subsequent subsections as they depend on the experiments. The traffic handling is different for the three transport mechanisms unicast, traditional IPMC, and BIER. BIER is considered with subdomains which are explained in Section VI-A2. The impact of the three transport mechanisms is quantified by the overall network load and link loads. Both metrics are explained in Section VI-A3. Since load heavily depends on packet sizes we discuss them in Section VI-A4. Finally, we explain the investigated factors in Section VI-A5 which have an influence on the performance results.

2) BIER Clustering: We recap our findings from Sections IV and V, and describe how we configured BIER for the evaluations.

BIER without subdomains cannot support arbitrary topology sizes without extensive headers. BIER with subdomains supports large topologies but its efficiency heavily depends on the subdomain clustering (see Section IV-A). Therefore, we designed an integer-linear program (ILP) to find optimal subdomain clusterings (see Section IV-D) in arbitrary topologies and presented optimal BIER clusterings for selected topologies (see Section IV-C). However, the ILP can compute clusterings only in small networks due to runtime restrictions (see Section V-B). Therefore, we designed a heuristic for that purpose (see Section IV-E) and showed that its results are reasonably close to results from the ILP (see Section V-C).

For all following evaluations we consider only BIER forwarding with subdomains. On random topologies we compute the subdomain clustering with the proposed heuristic from Section IV-E. For selected topologies, i.e., ring, line and binary tree, we leverage the presented optimal clustering approaches from Section IV-C.

If not stated otherwise, we assume in our studies for BIER a bitstring size of $b = 256$ bits because that value must be supported by all BIER-capable equipment. Thus, b is also the maximum number of BFERs in subdomains. We assume that all nodes are BFERs. That means, when networks have more than b nodes, the nodes are partitioned into a minimum number of $s = \lceil \frac{n}{b} \rceil$ subdomains.

3) Metrics: We utilized two performance metrics in the simulations: overall traffic load and link load. We describe them in the following.

a) Overall Traffic Load: In Section VI-B we evaluate the overall traffic load. The overall load is the accumulated number of bytes sent in an experiment over any link in the network

to distribute the packets from each source to all receivers. This value obviously depends on the transport mechanism. To quantify the traffic savings potential of IPMC and BIER we relate their overall load for a specific traffic scenario to the one of unicast and to each other. For all evaluations, packets follow shortest path trees based on the hop count metric.

b) Link Load: Traffic load is not equally distributed over all links. Central links tend to have higher load than others so that they may profit more from traffic load reduction through multicast. Therefore, we study link load reduction on links in Section VI-C. To that end, we count packets carried over specific links instead of bytes as this facilitates interpretation of the results.

4) Packet Sizes: Table 4 shows the total packet sizes for different transport mechanisms in byte (B). For unicast and IPMC traffic we assume a packet size of 520 B which is the average size of IP packets on the Internet [40]. For BIER packets we assume a total size of 564 B, i.e., 520 B payload including the IPMC header plus 44 B to respect the additional BIER header fields and a bitstring length of 256 bits. If a longer bitstring length is used, the additional bytes are added to the 564 B.

Transport mechanism	IPMC packet w/ payload (B)	BIER-X (B)	Total packet size (B)
Unicast	520	-	520
IPMC	520	-	520
BIER-256	520	44	564
BIER-512	520	76	596
BIER-1024	520	140	660
BIER-2048	520	268	788
BIER-4096	520	524	1044
BIER-8192	520	1036	1556

Tab. 4: Total packet sizes for different transport mechanisms in byte (B). BIER-X refers to a BIER header with a bitstring of X bits. Thus, the total BIER header size is $(\mathrm{X}/8$ bits$)$ plus 12 B for all other BIER header fields.

5) Investigated Factors: We investigate the following four factors. First, we consider different topology-types. That is, we selected line, ring and binary tree topologies to evaluate scenarios where distributing traffic with traditional IPMC or BIER has significant advantages due to the many shared paths of packets. Furthermore, we investigate random topologies with different average node degrees. This factor is relevant for all following evaluations.

Second, we evaluate all mechanisms on different network sizes to determine the scalability in large networks (see Section VI-B1). Third, we vary the size of multicast groups (see Section VI-B2), and thereby the number of receivers. That is, not every node in the network may necessarily be a subscriber which influences the efficiency of the transport mechanisms.

Finally, we investigate the impact of the BIER header size. On the one hand, small BIER headers add only little overhead, on the other hand large BIER headers reduce the number of extra copies needed to reach all subscribers in large networks. We study this tradeoff in Section VI-D.

We evaluate those factors by keeping other factors stable and varying the desired parameter. For example, we chose full multicast groups and change only the network size to determine its impact on the metric.

B. Reduction of Overall Traffic

We evaluate the potential for the reduction of overall traffic through multicast variants relative to unicast and compare the efficiency of BIER with the one of IPMC. To that end, we measure the number of transmitted bytes in the network to distribute a packet from all sources to all destinations (see Section VI-A3a). We first study the impact of network topology and size and then the impact of network topology and multicast group size.

1) Impact of Network Size: We evaluate the savings potential for overall traffic through multicast variants. To that end, we consider different network topologies and sizes and maximum multicast groups. That is, every node is a subscriber and receiver. We study IPMC vs. unicast, BIER vs. unicast, and BIER vs. IPMC.

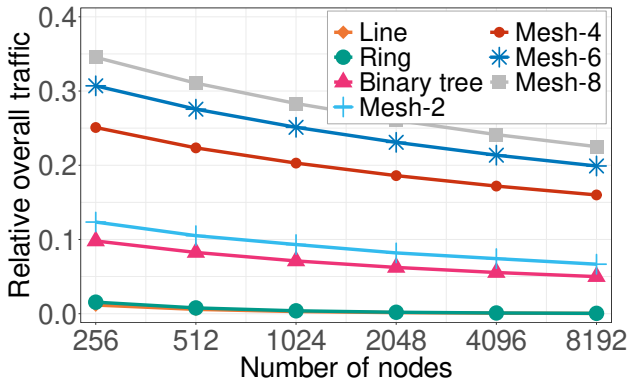
a) IPMC vs. Unicast: Figure 3(a) shows the overall traffic for IPMC relative to unicast for multiple network topologies depending on the network size. The IPMC traffic load decreases relative to the unicast traffic load with increasing network size. There is a large reduction potential in line and ring networks so that the IPMC traffic volume is less than 2% compared to the one of unicast. In perfect binary trees the traffic can be reduced to 10% for $n = 255$ nodes and to 5% for $n = 8191$ nodes. Random mesh networks have a lower reduction potential that decreases with increasing node degree.

We observe an obvious dependence of the traffic reduction potential of IPMC on the network topology. We show that it is $\frac{1}{l}$ in the presence of maximum multicast groups. Multicast requires $n-1$ hops to distribute a packet from one source to $n-1$ receivers as this is the number of links in any shortest-path tree. Thus, $n \cdot (n-1)$ hops are required to distribute a packet from each node to all other nodes. When the same is done with unicast, any source node $v \in \mathcal{V}$ sends a packet to any destination node $w \in \mathcal{V}$. This requires $|p(v,w)|$ hops per v/w pair, which is in sum

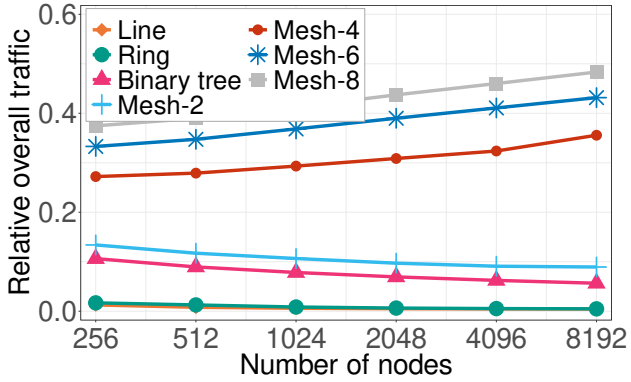
$$\sum_{v\in\mathcal{V}}\sum_{w\in\mathcal{V}} |p(v,w)| = n\cdot(n-1)\cdot \frac{\sum_{v\in\mathcal{V}}\sum_{w\in\mathcal{V}} |p(v,w)|}{n\cdot(n-1)} = n\cdot(n-1)\cdot l. \quad (7)$$

This follows that IPMC can reduce the overall traffic to $\frac{1}{l}$ compared to unicast. Lines and rings have by far the longest average path length and it strongly increases with increasing network size. In other topologies, average path lengths are clearly lower and increase slowly with the network size. The average path length correlates with the node degree and increases in the following topologies: mesh-8, mesh-6, mesh-4, mesh-2 and perfect binary trees.

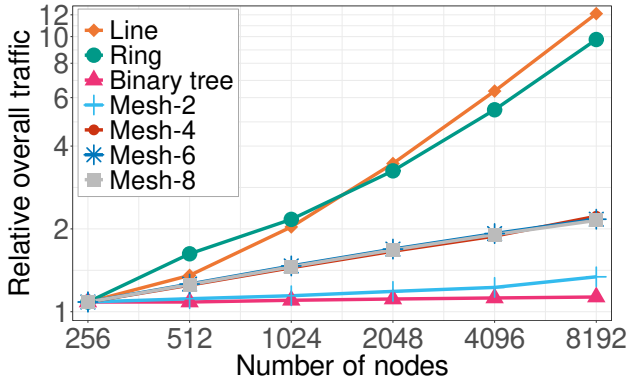
b) BIER vs. Unicast: Figure 3(b) presents the overall traffic load for BIER relative to unicast. The number of required subdomains increases with the network size and thereby



(a) IPMC relative to unicast. The curves for line and ring are almost identical.



(b) BIER relative to unicast. The curves for line and ring are almost identical.



(c) BIER relative to IPMC.

Fig. 3: Overall relative traffic load depending on the network size; every node sends one packet to all other nodes.

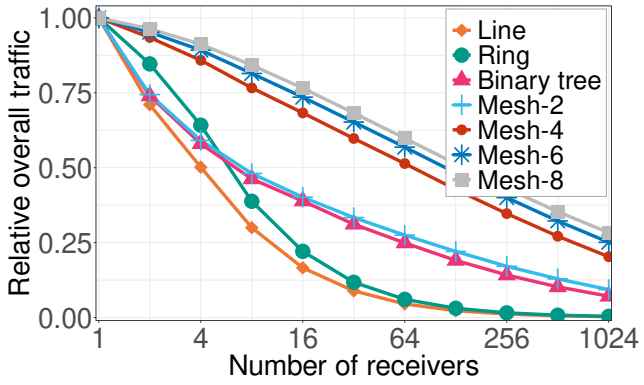
the number of BIER packets needed for these subdomains. We observe that BIER achieves the largest traffic reduction in lines and rings (around 97%), followed by perfect binary trees and mesh networks with node degree 2 (around 90%). Mesh networks with node degrees 4, 6, and 8 have a lower savings potential of around 50% and BIER's traffic savings potential relative to unicast decreases with increasing network size. The latter is different to IPMC (cf. Figure 3(a)). Thus, BIER can reduce the load of multicast traffic similarly well as IPMC, except in large, highly meshed networks in spite of well clustered subdomains.

c) BIER vs. IPMC: To compare BIER directly with IPMC, we consider the fraction of overall traffic load of BIER and the one of IPMC in Figure 3(c). In line and ring networks, BIER causes a multiple (3 - 11 times) of the traffic that occurs with IPMC if the number of subdomains is very large, i.e., 8, 16, and 32 for network sizes of 2048, 4096, and 8192 nodes. That is because BIER with subdomains causes redundant packet copies and the BIER header adds extra bytes which have to be transmitted. However, the previous evaluation shows that the savings compared to unicast are still enormous, i.e., more than 98%. In mesh networks with node degee 4, 6, and 8, the traffic load with BIER compared to IPMC increases about logarithmically with network the network size and roughly doubles the load with IPMC in very large networks. In perfect binary trees and mesh networks with node degree 2, the traffic load with BIER relative to IPMC also increases with network size, but BIER causes only 40% more traffic than IPMC in very large networks although up to 32 subdomains are supported.

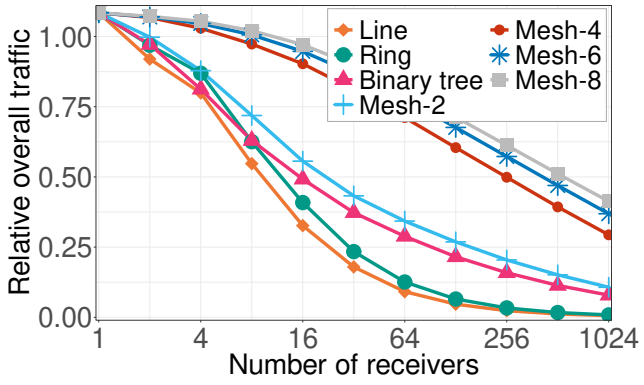
2) Impact of Multicast Group Size: We evaluate the influence of the multicast group size on the traffic reduction potential of multicast variants. We perform this study for different network topologies and a network size of $n = 1024$. Every node sends a packet to a random set of receivers. The set sizes are $r \in \{1, 3, 7, 15, 31, 63, 127, 255, 511, 1023\}$ large, for perfect binary trees the maximum number of receivers is $r = 1022$. As these are random experiments, we run each experiment 20 times to obtain very small confidence intervals that we omit in the figures for the sake of clarity.

a) IPMC vs. Unicast: Figure 4(a) shows the overall traffic load of IPMC relative to unicast. It decreases with increasing multicast group size. For small multicast group sizes, IPMC can reduce the overall traffic only by little. In line and ring networks, large traffic savings are achieved already for small multicast groups, followed by perfect binary trees and random meshes with node degree 2. Random meshes with node degree 4, 6, and 8 require rather large multicast groups to provide a substantial savings potential.

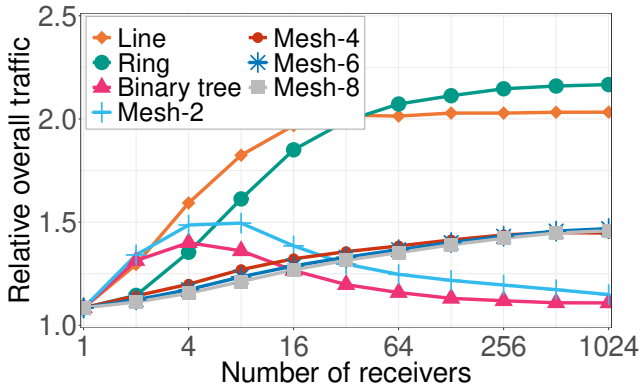
b) BIER vs. Unicast: Figure 4(b) shows the overall traffic load for BIER vs. unicast. The figure looks similar to Figure 4(a), but all lines are slightly higher than with IPMC as the BIER header induces additional overhead. In particular, for a small number of receivers the overall traffic load with BIER relative to unicast is larger than 1 because the overhead generated by the BIER header is larger than the overhead saved



(a) IPMC relative to unicast.



(b) BIER relative to unicast.



(c) BIER relative to IPMC.

Fig. 4: Overall traffic load depending on the size of the multicast group; the networks are n = 1024 nodes large and every node sends one packet to the given number of random receivers.

through BIER in terms of saved packet copies.

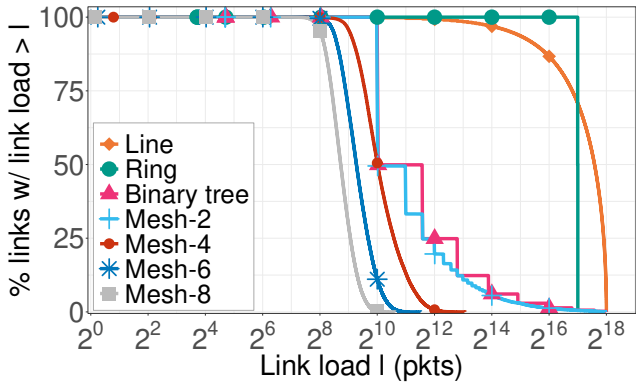
c) BIER vs. IPMC: Figure 4(c) shows the overall traffic load with BIER compared to the one of IPMC. For a single receiver, BIER and IPMC are almost equally efficient as BIER does not send any redundant packets and the BIER header adds only a small overhead. . When the number of receivers increases, the overhead of BIER increases as more redundant BIER copies are sent. The values for $r = 1024$ receivers are same as the values for network size $n = 1024$ in Figure 3(c). Apart from that, BIER's overhead compared to IPMC depends on the network topology. For line and ring topologies, the overhead of BIER relative to IPMC increases up to a multicast group size of 64 receivers and remains constant afterwards. For mesh networks with node degree 4, 6, and 8, the overhead increases logarithmically with increasing number of subscribers. And for line and ring networks, the overhead decreases when the number of subscribers is 8 receivers or more.

C. Avoidance of Heavily Loaded Links

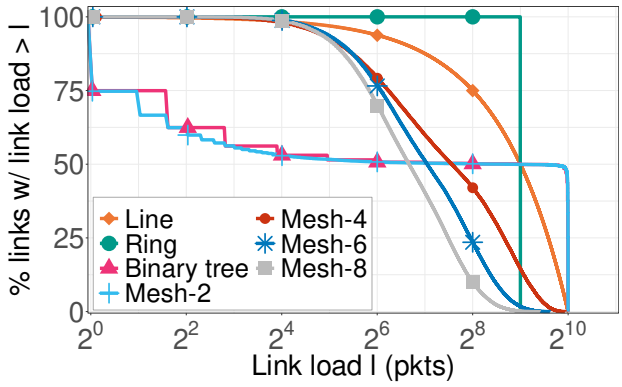
For some network topologies, the savings potential through multicast is only moderate. However, traffic is not equally distributed over all links of a network as central links tend to carry more traffic. We show that multicast variants can greatly reduce the load on those links compared to unicast. We consider again maximum multicast groups and networks with $n = 1024$ nodes, $n = 1023$ for perfect binary trees, where every node is subscriber and receiver. We count the number of bytes carried over each link and discuss the complementary cumulative distribution function (CCDF) of the link loads (see Section VI-A3b) for unicast, IPMC, and BIER. In case of mesh-d, the load information of multiple networks is integrated in a single CCDF.

1) Link Load Distribution with Unicast: The CCDF for link loads with unicast is illustrated in Figure 5(a). In line and ring networks, a large percentage of links carries a large number of packets. With rings, any link carries the same number of packets due to symmetry. In perfect binary trees the number of packets per link is clearly lower than in lines and rings. Half of the links has only a load of 1022 packets, those are adjacent to the leaves. There are also a few links with a very high link load, those are links close the root. Random mesh networks with a node degree 2 have a similar CCDF as perfect binary trees. The random mesh networks with a node degree of 4, 6, and 8 have increasingly lower link loads and less variation regarding link loads.

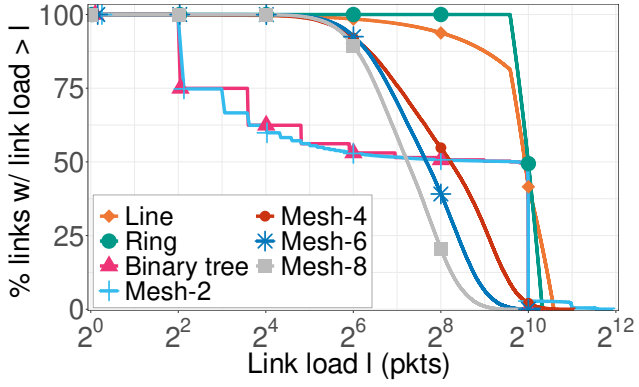
2) Link Load Distribution with IPMC: Figure 5(b) shows the CCDF for link load with IPMC. The upper limit is now 1023 packets. Again, many links in lines and rings carry a large number of packets. In perfect binary trees and random mesh networks with node degree 2, 25% of the links have a very low load while 50% have a high load. In perfect binary trees, those are links close to the leaves and to the root, respectively. Random meshes with node degree 4, 6, and 8 reveal again a load continuum but it is at a lower load level compared to unicast. The x-scales in Figures 5(a) and 5(b) are



(a) Unicast.



(b) IPMC.



(c) BIER.

Fig. 5: Complementary cumulative distribution function (CCDF) for link loads in networks with 1024 nodes; every node sends one packet to all other nodes.

different. This suggests that there are many links for which IPMC decreases the traffic load by orders of magnitude. Thus, IPMC avoids in particular heavily loaded links compared to unicast.

3) Link Load Distribution with BIER: Figure 5(c) shows the CCDF of link loads with BIER. It looks similar to the one of IPMC in that the link load is mostly limited to 1023 packets. However, some links in line and ring networks have larger loads up to around 1600 packets, and a very few links in mesh networks with node degree 2 have loads of up to 3976 packets, i.e., roughly the maximum link load for multicast multiplied by the number of subdomains s. Mesh networks with a node degree of 4, 6, and 8 generally lead to lower link loads as their traffic is not concentrated on a few links.

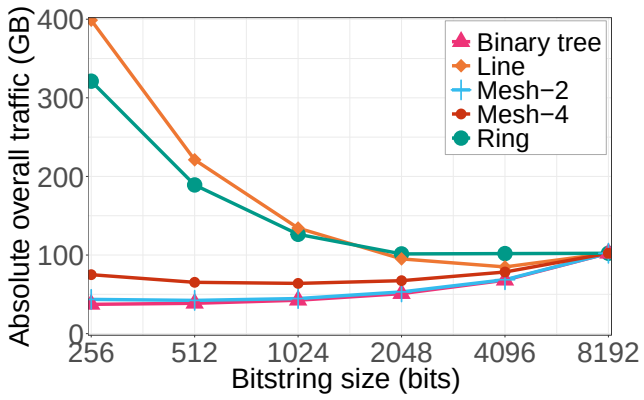
The most important finding is that BIER also avoids very high loads on links compared to unicast. Only a very few links experience substantially higher link loads than with IPMC. Thus, BIER efficiently avoids heavily loaded links in a similar way as IPMC.

D. Impact of BIER Header Size

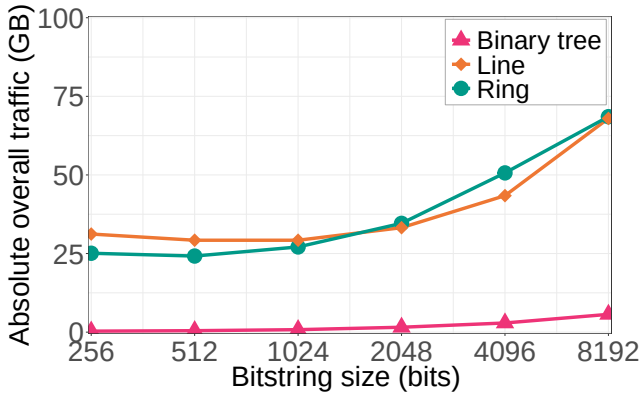
On the one hand, BIER largely avoids redundant packets over links, on the other hand BIER causes additional header overhead. There is an obvious tradeoff regarding header size: small bitstrings add only little header overhead, but require many redundant packets in large network, large bitstrings add lots of header overhead, but require only a few redundant packets in large networks. We expect an optimal header size in between. To study this effect, we first explain our methodology and then discuss experimental results.

1) Methodology: We consider networks with $n = 8192$ nodes, $n = 8191$ for perfect binary trees. We investigate different bitstring sizes of $b \in \{256, 512, 1024, 2048, 4096, 8192\}$ bits that require $\lceil \frac{n}{b} \rceil$ SDs. We measure the total overall traffic in bytes as performance metric as this captures the effect of the BIER header size. We omit mesh-6 and mesh-8 topologies because results were almost the same as for mesh-4.

2) Results: We first study maximum-size multicast groups and each participant sends a single packet. Figure 6(a) shows the overall traffic volume for different bitstring sizes. The ring and the line topology lead to a large traffic volume for small bitstring sizes b. This results from long paths of most of the packets replicated for the $\lceil \frac{n}{b} \rceil$ SDs. Larger bitstring sizes reduce the number of SDs and thereby the replicated packets as well as the traffic volume. The optimal bitstring size for the line is $b = 4096$ bits and the one for the ring is $b = 2048$ bits. Larger bitstring sizes add so much header overhead that the overall traffic increases again. Topologies with shorter paths like binary trees, mesh-2 or mesh-4 networks reveal a clearly lower traffic volume for small bitstring sizes than lines or rings. The optimal bitstring size is $b = 256$ bits for binary trees, it is $b = 512$ bits for mesh-2, and it is $b = 1024$ bits for mesh-4. However, suboptimal bitstring sizes between 256 and 2048 bits lead only to slightly larger traffic volumes. Thus, any of these bitstring sizes is suitable for typical network topologies.



(a) Maximum-size multicast groups.



(b) Multicast groups with 128 random receivers.

Fig. 6: Overall traffic load depending on the bitstring size in the BIER header in networks with $n = 8192$ nodes.

The findings slightly change when we consider smaller multicast groups, i.e., multicast groups consisting of 128 randomly selected nodes. The corresponding results are shown in Figure 6(b). First, the overall traffic volume is clearly decreased as the number of receivers is lower (1.56%). Further, the optimum bitstring sizes are smaller, namely $b = 512$ for the ring and $b = 1024$ for the line. Thus, the optimum bitstring size depends on the size of the multicast groups. Therefore, the bitstring size is hard to optimize for practical applications when the size of the multicast groups is not known a priori. However, if the multicast groups are small, a small bitstring is recommendable. This makes the application of subdomains and their optimization even more relevant.

VII. Impact of Single Link Failures

We have optimized BIER subdomains for failure-free forwarding. In case of link failures, rerouting occurs in IP networks and then traffic is diverted around failed links. As a consequence, individual link loads and overall traffic load may increase. BIER with subdomains optimized for failure-free routing may lead to an even larger traffic increase than IPMC forwarding. Therefore, BIER may require more backup capacity than IPMC. We investigate these issues in the following. We first explain our methodology. Then, we perform simulations to study the overall traffic load and maximum link loads in case of single link failures, as well as the overall backup capacity required to accommodate rerouted traffic.

A. Methodology

Single link failures may partition a network topology. Then multicast groups are also partitioned into subgroups that cannot reach each other anymore. This can be avoided in resilient networks with 2-link-connected topologies and rerouting after failure detection. Thereby, end-to-end connectivity is not impaired so that participants of a multicast group can still reach each other. As a consequence, we consider only 2-link-connected topologies in this context, i.e., networks which are still connected after any single link failure. As a consequence, we do not consider lines and binary trees as they may be partitioned through single link failures. Rings are 2-link-connected by definition. We reuse the mesh-$\{4,6,8\}$ topologies from Section V-A which were chosen for the entire study such that they are 2-link-connected.

We consider networks with 1024 nodes and a bitstring with $b = 256$ bits. We optimize the subdomains for the failure-free case because it is the most common network state. That is, we use the heuristic clustering algorithms from Section IV-E for mesh-d topologies and the optimal clustering algorithm from Section IV-C for the ring topology. We compute subdomains based on the intact topology and evaluate BIER's efficiency when links in the network fail. We assume again a full multicast group and each participant sends a single packet to all other participants. We compute the effect of all single link failures for the mentioned topologies. That means, we remove the failed link from the topology, calculate new shortest paths, and compute the overall traffic load (see Section VII-B) and the maximum load increase on links (see Section VII-C); thereby, the subdomains remain unchanged. As mesh-d topologies are random, we report averaged results for them from 10 different topology samples.

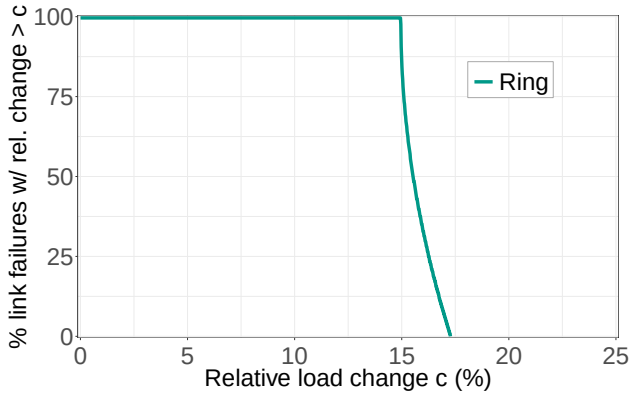
In our experiments we count number of packets carried over links. When we extend the single sent packets to flows, we obtain observed rates which are proportional to the numbers of counted packets. To be more intuitive, we sometimes talk about rates and required capacities rather than counted packets, in particular when it comes to backup resources.

B. Overall Traffic Load

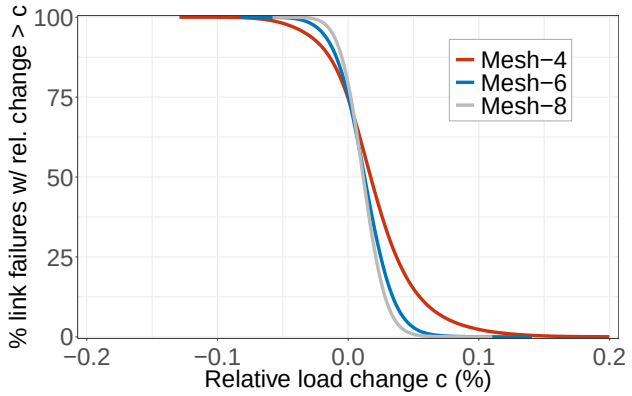
Traffic rerouting due to link failures possibly leads to longer paths, which may increase the overall link load in the network. Thus, we quantify the impact of single link failures on the overall traffic load (Equation 1) and compare it to the failure-free case both for IPMC and for BIER.

As the multicast groups in our experiments contain every node in the network, the overall traffic load for IPMC is $n \cdot (n-1)$ packets, no matter if a link fails. This is due to the fact that n packets are each forwarded along a single shortest path tree, and each shortest path tree consists of $n-1$ hops.

Thus, the traffic load does not increase with IPMC in case of single link failures.



(a) Ring.

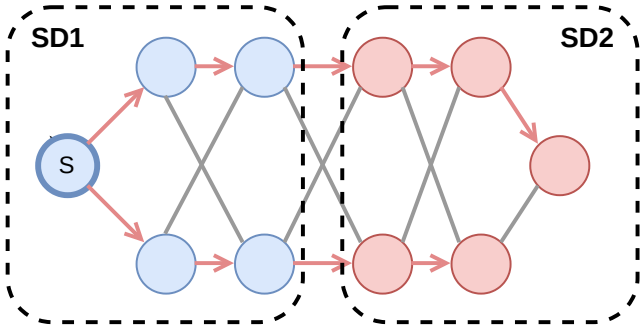


(b) Mesh-d.

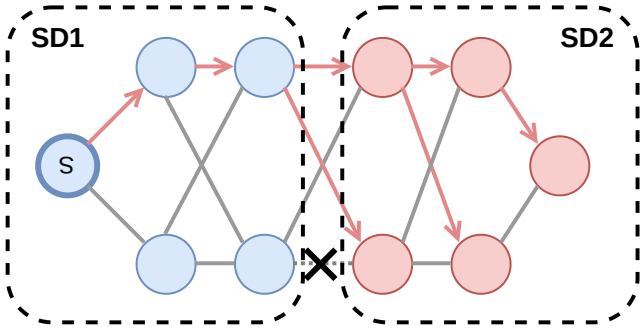
Fig. 7: BIER with single link failures – CCDFs of relative overall traffic change compared to the failure-free case, accumulated over all single link failures. Experiments are conducted in networks with $n = 1024$ nodes, every node sends a packet to every other node.

This is different with BIER. With BIER, $\lceil \frac{1024}{256} \rceil = 4$ packet copies, one for each subdomain, are forwarded over shortest path trees which consist of fewer hops than $n - 1$. However, their overall number of hops may change when traffic is rerouted. Therefore, we evaluate the change of overall traffic load with BIER for all single link failures. Figures 7(a) and 7(b) show CCDFs of relative overall traffic changes accumulated over all single link failures. We first discuss Figure 7(a) for a ring network. The overall traffic load rises between 15% and 17.3% depending on the position of the failed link. We explain this large increase as follows. Between any two nodes, there are exactly two paths in a ring network and the paths may have significantly different length. If the shorter path fails, traffic is rerouted over the longer path. This causes path stretch and leads to the observed increase in overall traffic load.

We now study mesh-d topologies for which the CCDF of the change in overall traffic load is presented in Figure 7(b). The increase in overall traffic load is bounded by 0.2%. We explain this as follows. In meshed networks with a node degree between 4 and 8, multiple paths exist between any two nodes and their lengths are likely to be similar. If the shortest path fails, another path with similar length is mostly available, which hardly increases the overall traffic load.



(a) Failure-free case.



(b) Single link failure.

Fig. 8: Example network with two BIER subdomains. In case of the indicated link failure, the adapted shortest path tree for the nodes in SD2 contains fewer hops than in the failure-free case, which reduces the traffic load.

We further observe that in 75% of all single link failures, the overall traffic load increases but in 25% the overall load decreases. This observation does not seem intuitive as the shortest path length for any pair of nodes remains unchanged or increases in case of a link failure. Nevertheless, the load may decrease as the shortest path tree towards the nodes in a subdomain may be more compact after rerouting. We illustrate this claim by the example in Figures 8(a) and 8(b). They show a network partitioned into two subdomains, SD1 and SD2. The shortest path tree starting in node S towards all nodes in SD2 contains two hops less in case of the considered link failure (Figure 8(b)) than under failure-free conditions (Figure 8(a)). This apparently more favorable path layout cannot be utilized under failure-free conditions because BIER traffic is always forwarded according to the paths in the underlay.

C. Maximum Load Increase on Links

When traffic is rerouted over another path, the traffic load on the corresponding links increases. We record for each link

the maximum load increase observed for any single link failure as this constitutes the required backup capacity for this link.

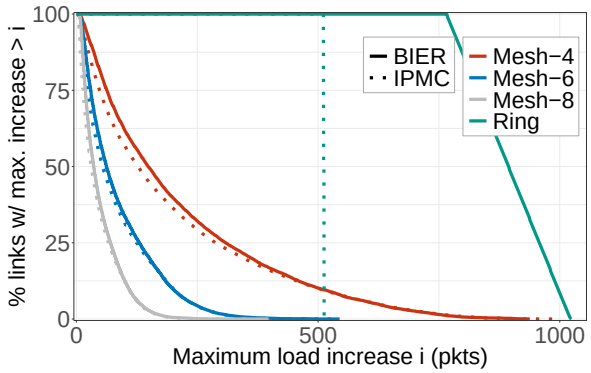


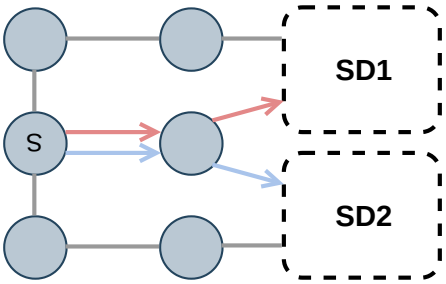
Fig. 9: CCDFs of maximum link load increases for single link failures. Experiments are conducted in networks with $n = 1024$ nodes, every node sends a packet to every other node.

Figure 9 shows the CCDFs of the maximum load increases for all links. In ring networks, all links experience up to 512 more packets with IPMC in case of link failures. In contrast with BIER, links carry between 768 and 1024 more packets. This is because multiple redundant BIER packets may be affected by the failure and are redirected. Therefore, BIER requires substantially more backup capacity in rings than IPMC and the exact amount depends on the location of a link within its subdomain. In mesh-d networks, the CCDF is almost a continuum. In networks with larger node degree, links require less backup than in networks with smaller node degree. This is due to shorter paths and less affected traffic, shorter backup paths, and better traffic distribution in case of link failures. Most notably, BIER causes about the same maximum load increases as IPMC although BIER requires more capacity than IPMC under failure-free conditions. We explain this fact by an example. Figure 10(a) shows a link carrying redundant BIER packets to two different subdomains. When that link fails, the traffic is redirected over different paths to the subdomains. IPMC would save a packet copy in the failure-free case, but it results into the same traffic distribution in this particular example.

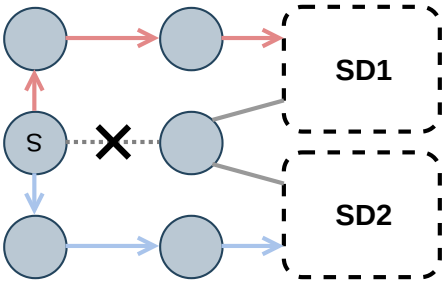
D. Overall Backup Capacity

We sum up link capacities for a network needed to carry the considered traffic for failure-free conditions on the one hand (capacity w/o backup) and for all single link failures on the other hand (capacity w/ backup). The difference is the absolute backup capacity. Table 5 compiles them for BIER and IPMC in mesh-d and ring topologies. The relative backup capacity is the ratio between absolute backup capacity and capacity w/o backup.

The results show that IPMC require 100% relative backup capacities for rings, but only 77%, 49%, and 36% for mesh-4, mesh-6, and mesh-8 networks. In contrast, BIER needs 176% backup capacity for rings, and 62%, 38%, and 29% for mesh-4, mesh-6, and mesh-8 networks. This is less than for IPMC,



(a) Failure-free case: two redundant packets are delivered over a link to two different subdomains.



(b) Single link failure: the two packets are redirected over different backup paths.

Fig. 10: Example network with two BIER subdomains. Redundant BIER packets for different subdomains are redirected over different paths.

	Metric	Ring	Mesh-4	Mesh-6	Mesh-8
IPMC	Cap. w/o backup	1047552	1047552	1047552	1047552
	Cap. w/ backup	2095104	1857633	1559138	1422539
	Abs. backup cap.	1047552	810081	511586	374987
	Rel. backup cap.	1.00	0.77	0.49	0.36
BIER	Cap. w/o backup	1051129	1395817	1418709	1406915
	Cap. w/ backup	2881534	2263645	1962557	1813694
	Abs. backup cap.	1830405	867828	543848	406779
	Rel. backup cap.	1.74	0.62	0.38	0.29
BIER / IPMC	Fraction w/o backup	1.003	1.33	1.35	1.34
	Fraction w/ backup	1.375	1.22	1.26	1.27

Tab. 5: Overall capacity w/ and w/o backup as well as absolute and relative backup capacity for IPMC and BIER. Capacities are given in packets.

which seems surprising, but there is a simple explanation. BIER requires more capacity w/o backup than IPMC, but only little more backup capacity than IPMC. As a consequence, BIER's relative backup capacity is lower than the one for IPMC.

Below the line, BIER does not lead to excessive backup capacity demands when BIER subdomains are optimized for failure-free scenarios. The relative backup capacity is even lower than with IPMC. The ring network is an exception, but also IPMC requires lots of capacity in rings.

VIII. Conclusion

BIER is a novel forwarding paradigm to carry IP multicast (IPMC) traffic within so-called BIER domains. It is more scalable than IPMC because core nodes remain unaware of individual multicast groups. A problem arises for large BIER domains where subdomains need to be defined to make all egress nodes addressable. When an IPMC packet is distributed via a BIER domain, a separate BIER packet is needed for each subdomain that has a receiver for the IPMC packet. This leads to redundant packets and we showed that their number almost equals the number of subdomains if multicast groups are about 3 times larger than the number of subdomains. These redundant packets can significantly degrade BIER's ability to efficiently carry multicast traffic.

We argued that an appropriate choice of the subdomains can mitigate that effect when multiple BIER packets are sent to different regions of a network. Therefore, we defined the BIER clustering problem and proposed several algorithms to cluster a BIER domain into appropriate subdomains. We compared the runtime and quality of these algorithms, and showed that optimization of subdomains can greatly reduce the resulting overall traffic compared to random subdomains.

We evaluated and compared the ability of IPMC and BIER to reduce traffic load for multicast traffic relative to unicast transmission in different network topologies. It depends on the average path length in the network. IPMC can save lots of traffic in line and ring networks, in binary trees and in mesh networks with a low node degree. In mesh networks with larger node degree the traffic savings potential is smaller. It also depends on the network size. As BIER possibly sends redundant packets in large domains, its ability to reduce traffic load diminishes compared to IPMC. This also depends on network topology and size. In large networks with 8192 nodes and subdomain sizes of 256 nodes, BIER causes only moderate extra traffic compared to IPMC in binary trees and mesh networks with small node degrees. In contrast, it produces 10-12 times more traffic than IPMC in lines and rings, but the traffic savings potential of BIER is still very large in these topologies ($\sim$ 98%). In mesh networks with larger node degrees BIER doubles the overall traffic compared to IPMC and also the traffic savings potential is clearly reduced. These findings hold for maximum multicast groups. In smaller multicast groups the traffic savings potential of IPMC and BIER relative to unicast transmission is lower. While unicast causes enormous traffic loads on some links, both IPMC and BIER decrease such loads by orders of magnitude. The residual load on these links is higher with BIER than with IPMC due to redundant packets, but it is still on a low level.

We showed that there is an optimum size for the BIER header which depends on the network topology and on the size of the multicast groups. When multicast groups are rather small, small BIER headers are optimal, which makes the use of subdomains and their optimum selection more relevant.

We investigated the impact of single link failures on BIER domains with optimized subdomains. Rerouting causes only little more traffic load and the backup capacity needed for BIER networks is only little more tha the one of pure IPMC networks.

Below the line, subdomains are a good means to scale BIER to large networks, but they need to be carefully chosen to minimize extra traffic due to redundant packets.

Further studies may improve BIER clustering algorithms with regard to quality. They may also consider alternate optimization goals such as the ability to take advantage of overlapping subdomains for known multicast groups. Furthermore, scaling BIER-TE is a related problem but it requires different approaches.

References

[1] N. K. Nainar, R. Asati, M. Chen, X. Xu, A. Dolganow, T. Przygienda, A. Gulko, D. Robinson, V. Arya, and C. Bestler, *BIER Use Cases*, https://datatracker.ietf.org/doc/draft-ietf-bier-use-cases/12/, Sep. 2020.

[2] I. Wijnands *et al.*, *RFC 8279: Multicast Using Bit Index Explicit Replication (BIER)*, https://datatracker.ietf.org/doc/rfc8279/, Nov. 2017.

[3] S. Islam *et al.*, "A Survey on Multicasting in Software-Defined Networking," *IEEE Communications Surveys Tutorials (COMST)*, vol. 20, 2018.

[4] Z. Al-Saeed *et al.*, "Multicasting in Software Defined Networks: A Comprehensive Survey," *Journal of Network and Computer Applications (JNCA)*, vol. 104, 2018.

[5] M. Shahbaz *et al.*, "Elmo: Source Routed Multicast for Public Clouds," in *ACM SIGCOMM*, 2019.

[6] A. Iyer *et al.*, "Avalanche: Data Center Multicast using Software Defined Networking," in *International Conference on Communication Systems and Networks*, 2014.

[7] W. Cui *et al.*, "Scalable and Load-Balanced Data Center Multicast," in *IEEE GLOBECOM*, 2015.

[8] X. Zhang *et al.*, "A Centralized Optimization Solution for Application Layer Multicast Tree," *IEEE Transactions on Network and Service Management (TNSM)*, vol. 14, 2017.

[9] K. Mokhtarian *et al.*, "Minimum-delay multicast algorithms for mesh overlays," *IEEE/ACM Transactions on Networking*, vol. 23, 2015.

[10] X. Li *et al.*, "Scaling IP Multicast on Datacenter Topologies," in *ACM CoNEXT*, 2013.

[11] M. A. Kaafar *et al.*, "A Locating-First Approach for Scalable Overlay Multicast," in *IEEE INFOCOM*, 2006.

[12] J. Rückert *et al.*, "Software-Defined Multicast for Over-the-Top and Overlay-based Live Streaming in ISP Networks," *Journal of Network and Systems Management (JNSM)*, vol. 23, 2015.

[13] J. Rueckert *et al.*, "Flexible, Efficient, and Scalable Software-Defined Over-the-Top Multicast for ISP Environments With DynSdm," *IEEE Transactions on Network and Service Management (TNSM)*, vol. 13, 2016.

[14] Y.-D. Lin *et al.*, "Scalable Multicasting with Multiple Shared Trees in Software Defined Networking," *Journal of Network and Computer Applications (JNCA)*, vol. 78, 2017.

[15] M. J. Reed *et al.*, "Stateless Multicast Switching in Software Defined Networks," in *IEEE International Conference on Communications (ICC)*, 2016.

[16] S.-H. Shen, "Efficient SVC Multicast Streaming for Video Conferencing With SDN Control," *IEEE Transactions on Network and Service Management (TNSM)*, vol. 16, 2019.

[17] T. Humernbrum *et al.*, "Towards Efficient Multicast Communication in Software-Defined Networks," in *IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2016.

[18] W. K. Jia *et al.*, "A Unified Unicast and Multicast Routing and Forwarding Algorithm for Software-Defined Datacenter Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 31, 2013.

[19] C. A. S. Oliveira *et al.*, "Steiner Trees and Multicast," *Mathematical Aspects of Network Routing Optimization*, vol. 53, 2011.

[20] L. H. Huang *et al.*, "Scalable and Bandwidth-Efficient Multicast for Software-Defined Networks," in *IEEE GLOBECOM*, 2014.

[21] S. Zhou *et al.*, "Cost-Efficient and Scalable Multicast Tree in Software Defined Networking," in *Algorithms and Architectures for Parallel Processing*, 2015.

[22] Z. Hu *et al.*, "Multicast Routing with Uncertain Sources in Software-Defined Network," in *IEEE/ACM International Symposium on Quality of Service (IWQoS)*, 2016.

[23] J.-R. Jiang *et al.*, "Constructing Multiple Steiner Trees for Software-Defined Networking Multicast," in *Conference on Future Internet Technologies*, 2016.

[24] B. Ren *et al.*, "The Packing Problem of Uncertain Multicasts," *Concurrency and Computation: Practice and Experience*, vol. 29, 2017.

[25] S.-H. Shen *et al.*, "Reliable Multicast Routing for Software-Defined Networks," in *IEEE INFOCOM*, 2015.

[26] A. Giorgetti *et al.*, "First Demonstration of SDN-based Bit Index Explicit Replication (BIER) Multicasting," in *IEEE European Conference on Networks and Communications (EuCNC)*, 2017.

[27] ——, "Bit Index Explicit Replication (BIER) Multicasting in Transport Networks," in *International Conference on Optical Network Design and Modeling (ONDM)*, 2017.

[28] D. Merling *et al.*, "P4-Based Implementation of BIER and BIER-FRR for Scalable and Resilient Multicast," *Journal of Network and Computer Applications (JNCA)*, vol. 169, 2020.

[29] ——, "Hardware-Based Evaluation of Scalable and Resilient Multicast With BIER in P4," *IEEE Access*, vol. 9, 2021.

[30] H. Chen, M. McBride, S. Lindner, M. Menth, A. Wang, G. Mishra, Y. Liu, Y. Fan, L. Liu, and X. Liu, *BIER Fast ReRoute*, https://tools.ietf.org/html/draft-ietf-bier-frr, Jul. 2022.

[31] Y. Desmouceaux *et al.*, "Reliable Multicast with B.I.E.R.," *Journal of Communications and Networks*, vol. 20, 2018.

[32] ——, "Reliable BIER with Peer Caching," *IEEE Transactions on Network and Service Management (TNSM)*, vol. 16, 2019.

[33] T. Eckert *et al.*, *Tree Engineering for Bit Index Explicit Replication (BIER-TE)*, https://datatracker.ietf.org/doc/html/draft-ietf-bier-te-arch, Jul. 2021.

[34] W. Braun *et al.*, "Performance Comparison of Resilience Mechanisms for Stateless Multicast using BIER," in *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2017.

[35] G. Karypis *et al.*, "A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs," *SIAM Journal on Scientific Computing (SISC)*, vol. 20, 1998.

[36] R. Diekmann *et al.*, "Shape-Optimized Mesh Partitioning and Load Balancing for Parallel Adaptive FEM," *Parallel Computing*, vol. 26, 2000.

[37] S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, 1982.

[38] A. Medina *et al.*, "BRITE: An Approach to Universal Topology Generation," in *International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2001.

[39] B. M. Waxman, "Routing of Multipoint Connections," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 6, 1988.

[40] F. Liu *et al.*, "The packet size distribution patterns of the typical internet applications," in *IEEE International Conference on Network Infrastructure and Digital Content*, 2012, pp. 325–332.

Daniel Merling is a Ph. D. student at the chair of communication networks of Prof. Dr. habil. Michael Menth at the Eberhard Karls University Tuebingen, Germany. There he obtained his master's degree in 2017 and afterwards, became part of the communication networks research group. His area of expertise include software-defined networking, scalability, P4, routing and resilience issues, multicast and congestion management.

Thomas Stüber is a Ph.D. student at the chair of communication networks of Prof. Dr. habil. Michael Menth at the Eberhard Karls University Tuebingen, Germany. He obtained his master's degree in 2018 and afterwards, became part of the communication networks research group. His research interests include Time-Sensitive Networking (TSN), scheduling, performance evaluation, and operations research.

Michael Menth (Senior Member, IEEE) is professor at the Department of Computer Science at the University of Tuebingen/Germany and chairholder of Communication Networks since 2010. He studied, worked, and obtained diploma (1998), PhD (2004), and habilitation (2010) degrees at the universities of Austin/Texas, Ulm/Germany, and Wuerzburg/Germany. His special interests are performance analysis and optimization of communication networks, resilience and routing issues, as well as resource and congestion management. His recent research focus is on network softwarization, in particular P4-based data plane programming, Time-Sensitive Networking (TSN), Internet of Things, and Internet protocols. Dr. Menth contributes to standardization bodies, mainly to the IETF.