Optimally DoS Resistant P2P Topologies for Live Multimedia Streaming

Michael Brinkmeier, Günter Schäfer, *Member*, *IEEE*, and Thorsten Strufe, *Member*, *IEEE*

Abstract—Using a peer-to-peer approach for live multimedia streaming applications offers the promise to obtain a highly scalable, decentralized, and robust distribution service. When constructing streaming topologies, however, specific care has to be taken in order to ensure that quality of service requirements in terms of delay, jitter, packet loss, and stability against deliberate denial of service attacks are met. In this paper, we concentrate on the latter requirement of stability against denial-of-service attacks. We present an analytical model to assess the stability of overlay streaming topologies and describe attack strategies. Building on this, we describe topologies, which are optimally stable toward perfect attacks based on global knowledge, and give a mathematical proof of their optimality. The formal construction and analysis of these topologies using global knowledge lead us to strategies for distributed procedures, which are able to construct resilient topologies in scenarios, where global knowledge can not be gathered. Experimental results show that the topologies created in such a real-world scenario are close to optimally stable toward perfect denial of service attacks.

Index Terms—Reliability, fault resilience, attack resilience, media streaming, peer-to-peer, overlay.

1 Introduction

Due to its scalability, cooperative streaming, sometimes called application layer multicast (ALM) [1], has become an increasingly interesting system architecture for live content distribution over the last years. These systems use the resources of end hosts and integrate participants as service proxies for other subscribers. While client-server solutions introduce a bottleneck and single point of failure at the server, cooperative streaming systems gain additional bandwidth and backup resources as the number of participants increases. As content is forwarded to the receivers by other participants of the system, each subscriber is dependent on all preceding participants in the overlay path to the original source. In consequence, participants with many successors, which are topologically close to the source, have a higher relevance to the overall system than participants near the leaves of the multicast trees that have fewer successors.

Systems designed for an application layer multicast are commonly classified into *push*- or *pull*-based approaches [2]. Push-based approaches create and maintain an explicit topology for the content dissemination. In pull-based approaches, each node explicitly requests the transfer of each part of the stream at other participating nodes. Systems of the latter category have to preload the requested parts well in advance of the play-out and their applicability to live streaming in consequence is limited, as this characteristic causes rather high delays [3], [4]. Push-based approaches again are commonly further classified into the categories *mesh-first* or *tree-first* [5], [6]. While mesh-first approaches create a management overlay first and set up the content dissemination topology using this mesh, tree-first approaches create the content dissemination topologies directly and use them for the distribution of management traffic, as well. However, using an overlay for streaming multimedia from a source to multiple receivers, each packet of the content is distributed along a set of links which connect all participating nodes. These links always form spanning trees, which are rooted at the source of the stream and concise of all participating nodes as either inner- or leaf-nodes. This characteristic applies for both push- and pull-based approaches, even though in pull-based approaches, these trees are neither created explicitly nor managed for multiple transmissions, and hence, possibly very short-lived.

In general, cooperative streaming systems, consisting of self-organizing hosts, show some inherent stability against node failures. This property stems from the domain of peer-to-peer systems, which is characterized by a high churn of node arrivals and node departures. In consequence, discovery and selection of alternative serving nodes as a fall-back strategy are an integral part of these systems. However, each reconnection of nodes due to dynamics in the system comes with the cost of additional messaging and topology management, leading to delays, jitter, and possible packet loss in the data transmission. Furthermore, while node failures and intentional departures usually happen at random locations in the system, a malicious attacker will try to gather information about the overlay and deliberately attack nodes which are important for the overall service. Therefore, in order to create systems which are also resilient to attacks, appropriate overlay topologies have to be constructed.

- *M. Brinkmeier is with the Automata and Formal Languages Group, Technische Universität Ilmenau, Postfach 100565, 98684 Ilmenau, Deutschland, Germany. E-mail: mbrinkme@tu-ilmenau.de.*
- *G. Schäfer is with the Telematics and Computer Networks Group, Technische Universität Ilmenau, Postfach 100565, 98684 Ilmenau, Germany. E-mail: guenter.schaefer@tu-ilmenau.de.*
- *T. Strufe is with the Networking and Security Team, EURECOM, 2229 route des crétes, BP 193, F-06560 Sophia-Antipolis cedex, France. E-mail: thorsten.strufe@eurecom.fr.*

Manuscript received 23 Apr. 2008; revised 17 Oct. 2008; accepted 18 Dec. 2008; published online 23 Dec. 2008..
Recommended for acceptance by C. Shahabi.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2008-04-0145.
Digital Object Identifier no. 10.1109/TPDS.2008.265.

In this paper, we focus on improving the stability of peer-to-peer-based distribution of multimedia streams against deliberate Denial of Service (DoS) attacks.

Intuitively, a couple of simple strategies to construct attack resilient overlay streaming topologies come to mind: The first idea is to keep the dependency of each node to other nodes low. This dependency is twofold: on the one hand, it is important for a node to minimize the amount of other nodes that it is dependent on, in order to minimize the chance of a predecessor leaving and, hence, to avoid quality degradations due to node departures. On the other hand, it needs to minimize its dependency to any single other node, in order to keep the impact of a leaving predecessor as low as possible. A second guideline is to keep the relevance of nodes balanced in order to avoid single nodes to become very important and a good target in consequence. A third important issue is to keep information about the topology as secret as possible, in order to make it hard for a malicious node to find good targets for attacking. However, this last requirement is difficult to meet in practice, as with knowledge about participants and about the distributed algorithm, it is always possible to make good estimations about the evolving topology.

In this paper, we make the following contributions: we provide an analytical model, which can be used to describe ALM systems. Using this model, we derive properties of topologies, which are optimally stable against attacks and give analytical proof of their optimality. These can serve as an upper bound to the resilience of topologies, both toward random node departures, caused by failure or churn, and DoS attacks on the system. The topologies additionally are characterized by the fact that they lead to a minimum deterioration of the quality of the delivered service, for *any number* of failing nodes. With knowledge about these properties, we are able to design a distributed procedure which creates close to optimally stable topologies in real scenarios, based on local knowledge only.

The remainder of the paper is organized as follows: In Section 2, we present previous approaches to the construction of stable streaming overlays. Subsequently, we define an analytical model for overlay streaming systems that we use to describe different damage functions as well as different attacker models in Section 3. We additionally define metrics for the attack and failure stability of ALM topologies, which we use to derive different types of attack strategies. Section 4 describes how optimal streaming topologies can be constructed and proves their optimal stability against perfect attacks based on global knowledge. Following this formal approach, in Section 5, we design a distributed procedure to construct stable topologies in real environments, based on local knowledge of the participating nodes only, followed by a simulative evaluation of the stability of different ALM topologies in Section 6. This performance comparison consists of three different types of topologies: optimally stable topologies, topologies that are created with the proposed distributed procedure, as well as topologies from previous ALM systems. In Section 7, we conclude our paper and give some directions for further work.

2 Related Work

Different approaches have been proposed so far in the literature to construct stable ALM topologies, which generally follow one of three strategies.

The first strategy is to increase the redundancy in the transmitted stream. It is frequently used on the application level through different Forward Error Correcting schemes. PRM [7], an extension of NICE [5], additionally tries to achieve resilience on the overlay level through randomly forwarding duplicates of content packets to randomly selected receiving nodes. This leads to some stability toward probabilistic packet loss and node failures, but does not protect against intentional attacks which aim at relevant nodes, and hence, disrupt the service close to the source.

A second strategy aims at reducing the amount of predecessors, which a node depends on. This is usually achieved through the construction of overlays which consist of low trees, rather than creating long paths between the source and the receivers. FatNemo [8], a derivative of NICE, constructs a fat tree of nodes: Nodes with good connectivity on high-bandwidth access links are located close to the original source and low-bandwidth nodes are placed further down in the multicast tree. FatNemo thus creates overlays of very broad and shallow trees. It additionally introduces fall-back strategies to cope with lost packets and mitigate the failure of cluster heads.

In order to allow for a graceful degradation in the events of departing or failing nodes, the third strategy is to lower the direct dependency between any two nodes, by transmitting the stream on different node-disjoint paths. This strategy leads to an increased vertex connectivity in the overlay. SplitStream [9], with the goal of achieving a fair balancing of the overall load, splits the stream into several *stripes* (partial streams) and creates topologies of a multitude of inner node disjoint trees. These are created using SCRIBE [10], an overlay publish-subscribe system based on Pastry [11], and each stripe is transmitted through paths of nodes which share the same node-ID prefix only, if bandwidth permits. These topologies show good stability properties as well, as the algorithm generally constructs topologies with high vertex connectivity. A drawback of SplitStream regarding the stability is the fact that single branches in the trees can grow to be quite long, thus introducing unnecessary and possibly harmful dependencies. Another set of systems aims at decreasing direct dependencies through the creation of directed acyclic graphs (DAGs). DagStream [12] attempts to increase the stability against node failures, by increasing the vertex connectivity of the streaming overlay. This is sought by connecting each node to a multitude of parent nodes, thus balancing the load of the service on them. However, in recent work [13], [14], we considered constructing stable topologies in a similar way, but had to realize, that this strategy quite frequently leads to topologies with a low vertex connectivity due to a distinct hour glass characteristic. Here, some nodes act as very relevant hubs by directly or indirectly serving many other nodes and become a very small minimum cut set. In consequence, their failure has a serious impact on succeeding nodes. Furthermore, as this class of systems arranges the nodes in a directed acyclic

graph, all systems consist of an algorithm to keep the topology loop free, which is based on the topological ordering. The information on the topological ordering, however, might be used by malicious nodes to gather information about the relevance of nodes, which greatly simplifies the detection of good targets for attacks. Magellan [15] combines the efforts of FatNemo and SplitStream to decrease the dependency between nodes and creates a fat tree using FatNemo for each stripe, while trying to keep the inner nodes of all trees disjoint.

An important problem of all proposed solutions is the fact that it is possible to retrieve information on the created topologies. In order to identify good targets, an attacker thus can gather information on the relevance of nodes or even "walk" the topologies to locate nodes which are placed near the source of the stream. Both for the creation of fat trees and the construction of a cycle free DAG explicit knowledge and decisive node characteristics are needed.

Summarizing this discussion, in order to create overlays, which are both stable to node failure and attack, topologies have to be constructed which are characterized by each node being dependent on as few nodes as possible, with a minimum dependency on a single predecessor and an even distribution of successors for all forwarding nodes.

3 Analytical Model

To model the streaming overlays, it suffices to focus on end-to-end links. The characteristics of the underlying network infrastructure do not need to be considered, as backbone and network routing decisions are not influenced by the systems and all overlay nodes additionally are able to establish a connection to any of the other overlay nodes. The abstraction from the underlying network topology leads to the inability to examine the behavior in circumstances of failing or attacked routers; however, the model suffices our needs, as we currently focus on end hosts only.

Generally, one source node s is the originator of the streaming content. All other joining nodes locate participants as potential sources, which have joined the service at an earlier time, select some of them as parents and offer the service of forwarding the content, in turn. This system of potential and selected neighbors can be modeled as an undirected graph $G = (V, E)$ with a finite set of n vertices $V = \{v_1, \ldots, v_n\}$, a data source $s \in V$ and the set of edges: $E \subseteq \{(u, v) | u, v \in V, u \neq v\}$. The multimedia content can be modeled as a packet stream: $\mathcal{S} = \{p_1, \ldots, p_p\}$ of p packets. All p packets can be replicated at each vertex and originate at the data source, the bitrate of the stream is denoted by R_0. Alternatively, the packet stream can be split into partial streams, with l sequences of k stripes: $\mathcal{S} = \{\{p_1^1, \ldots, p_k^1\}, \ldots, \{p_1^l, \ldots, p_k^l\}\}$. Each stripe, in consequence, has an average bitrate of R_0/k. The source s has a bandwidth capacity $c(s)$, which is C times the bitrate R_0 of the stream. Hence, it can deliver the whole stream C times, or $C \cdot k$ stripes, respectively, simultaneously. To model the overlay topologies, let

1. $d : E \to \mathbb{R}^+$ be a nonnegative edge length (e.g., the latency of the connection);
2. $c : V \to \mathbb{R}^+$ be the vertex capacity (bandwidth of the access link).

The topology control constructs k rooted spanning trees: $T_1 = (V, E_1), \ldots, T_k = (V, E_k)$ in G, preferably with a minimum total cost

$$\sum_{i=1}^{k} d(T_i) = \sum_{i=1}^{k} \sum_{e \in E_i} d(e),$$

constrained by the degree of each vertex $v \in V$ in all T_i being at most $c(v)$:

$$\sum_{i=1}^{k} \deg^{\mathrm{T_i}}(\mathrm{v}) \leq \mathrm{c(v)} \quad \text{for all v} \in \mathrm{V},$$

with s being the root in all trees. The sequence $\mathcal{T} = (T_1, \ldots, T_k)$ of the spanning trees is the streaming topology. Additionally, let $\mathbf{C}(n, k, C)$ be the class of all topologies with n nodes, k stripes, and a source capacity of C. The *depth* $\mathrm{depth(v)}$ of a vertex v in a rooted tree T is its distance from the root s. The ith *level* $L_i(T)$ of T is the set of all vertices of depth i and $\mathrm{depth}(\mathcal{T})$ is the maximum depth of all of the trees in the topology $\mathcal{T}$, i.e.,

$$\mathrm{depth}(\mathcal{T}) := \max\{\mathrm{depth}(T_i) \mid 1 \leq i \leq k\}.$$

For a node v in $\mathcal{T}$, $\mathrm{depth}(v)$ is the maximum depth of v in $\mathcal{T}$, i.e.,

$$\mathrm{depth}(v) := \max\{\mathrm{depth}^{T_i}(v) \mid 1 \leq i \leq k\}.$$

A *head* in a topology $\mathcal{T}$ is a direct successor of the source s. More precisely a *head of or in stripe* i is a direct successor of s in the ith stripe. Let $\mathcal{H}^i$ be the set of all heads in stripe i and $\mathcal{H}$ the set of all heads. Due to the capacity constraint of the source, we have $|\mathcal{H}| \leq C \cdot k$.

To characterize the importance of a participant, we define $\mathrm{succ}_i(v)$ as the set of successors of v in the spanning tree T_i (including v itself). For a set X of nodes, $\mathrm{succ}_i(X)$ is the set of all direct and indirect successors of a vertex in X, i.e.,

$$\mathrm{succ}_i(X) = \bigcup_{v} \in X \mathrm{succ}_i(v).$$

3.1 Attacks and Failures

Obviously, it is always possible to interrupt a multimedia stream by destroying the camera or the filmed scene. Strategies to secure the source of the stream like backup servers are conceivable. However, as we are concerned about constructing stable overlay topologies, we consider the source node s being hidden and assume that it cannot be the target of attacks. Hence, we only consider the failure of peers.

Usually, the vertex connectivity is used as a stability metric for topologies, as it gives the minimum set of nodes that have to be removed in order to split the system into two separate fragments, one being completely disconnected from the data source. However, as the multimedia streaming service already has to be rendered useless as soon as the amount lost packets exceeds a certain threshold, a different metric is needed.

In general, an attacker chooses a set X of nodes and forces them to stop forwarding packets. How this is achieved is not considered in the following. Hence, we may simply assume that the nodes fail completely. In terms

of our previously introduced model, the attacker enforces the removal of the nodes in X from each of the trees T_i of the current topology $\mathcal{T}$. Of course, this causes a loss of the ith stripe at each successors of a node in X.

In full generality, an *attacker* observes a topology $\mathcal{T}$ and, due to limited resources, wants to attack at most i nodes to achieve damage. In mathematical terms, an *attacker* or an *attack strategy* can be viewed as a map $\mathcal{A}$, assigning a set $\mathcal{A}(\mathcal{T}, i)$ with at most i elements to every topology $\mathcal{T}$ and every natural number i with $0 \leq i \leq n$.

This model is quite general and covers many situations. First of all, the model includes external attackers, which cause damage by selecting target nodes, and then perform DoS or similar attacks to remove them from the system. But the model even covers internal attackers. These introduce a specific number of *agents* into the system, which basically act as peers. Using incorrect information and system inherent mechanisms, these agents are brought into specific positions and simply stop relaying content at a certain time. However, on an abstract level, this approach is equivalent to an external attacker, which chooses certain nodes (those at the positions, at which his agents should be), and attacks them directly. Hence, internal and external attackers can be modeled.

Furthermore, it is possible to use *nondeterministic attackers*, by adding a probabilistic component to $\mathcal{A}$. In this case, the attacked set is chosen randomly using a distribution depending on the topology and the number of maximal attacked nodes. This type of attackers covers random failure and even churn (but neglecting the arrival of new nodes). In the first case, a set of at most i nodes is chosen randomly. In the second case, each node fails with a specific probability, representing its reliability.

3.2 Damage

In general, the attacker evaluates the success of its attack using a measure of caused damage. For this mean, we introduce two functions.

For each tree T_i, we define a function $a_i^{\mathcal{T}} : \mathcal{P}(\mathcal{V}) \to \mathbb{N}$, from the set $\mathcal{P}(\mathcal{V})$ of all subsets of V to the natural numbers, with

$$a_i^{\mathcal{T}}(X) := |\mathrm{succ}_i(X)|,$$

i.e., $a_i^{\mathcal{T}}(X)$ is the number of all successors of elements in X.

To obtain the total number of missing packets, we have to sum over all trees, i.e., $a^{\mathcal{T}} : \mathcal{P}(V) \to \mathbb{N}$ is given by $a^{\mathcal{T}}(X) := \sum_{i=1}^{k} a_i^{\mathcal{T}}(X)$.

In case of a node failure or deliberate node leave, the packets this node is not receiving are not expected by the concerned node. Hence, their loss does not reduce the quality of the service, and as it could not be prevented in any case, they are not counted as damage. Consequently, only packets depending on the failing nodes have to be counted, and not the $k \cdot |X|$ packets lost due to the failure of the nodes in X. Therefore, the damage caused by node failure is

$$f^{\mathcal{T}} : \mathcal{P}(V) \to \mathbb{N}$$

with

$$f^{\mathcal{T}}(X) := a^{\mathcal{T}}(X) - k \cdot |X|.$$

The additional function

$$\mathrm{inc}_X(v) = \text{The number of trees in which neither } v \text{ nor any of its predecessors has failed}$$

allows us to characterize the perceived quality of the service under the circumstance of failing nodes.

In case of the removal of nodes, a target has to be selected first. Both deliberate node leave or failure, as well as naive attacks lead to the removal of random nodes from the topology. A realistic attacker, however, will try to gather knowledge about the relevance and select specific nodes for removal. Different possibilities to gain information about the amount of successors of nodes are analyzing the distributed algorithm, observing parent and child nodes, or probing a large set of participants in order to gain knowledge about their neighboring relations. The perfect attack is based on global knowledge about the complete topology $\mathcal{T}$ and represents a worst-case stability metric for topologies, both with respect to attacks and node failures.

To evaluate potential goals of an attacker, we define two types of damage functions first.

The *global damage* expresses the overall packet loss rate in the whole system, inflicted by the removal of the node set X. It leads to quality degradation for all users and can be calculated by $a^{\mathcal{T}}(X)$.

The resulting attacker model for *global attack* is defined as:

Given a streaming topology $\mathcal{T}$ and the packet loss threshold $r_d \in (0, 1)$, find the smallest set X of removed nodes, such that

$$a^{\mathcal{T}}(X) > r_d \cdot k \cdot n.$$

In order to measure the worst-case stability of the topology, a perfect attacker with global knowledge is considered. Two conceivable strategies for such a perfect attacker are either selecting the minimum set of nodes to achieve a given global damage or selecting a set of nodes of a given size with maximum global damage. For the rest of this paper, we focus on measuring the stability of topologies through an attacker that maximizes the damage for increasing sets of attacked nodes.

3.3 Attack and Failure Stabilities

Each (deterministic) attacker $\mathcal{A}$ causes a specific damage on a topology $\mathcal{T}$, which is given by

$$a_{\mathcal{A}}^{\mathcal{T}}[i] := a^{\mathcal{T}}(\mathcal{A}(\mathcal{T}, \rangle)).$$

If $\mathcal{A}$ is a randomized attacker, it is sensible to consider the expected damage. Similar, $f_{\mathcal{A}}^{\mathcal{T}}[i]$ can be defined as the failure caused by the attacker $\mathcal{A}$. Using these numbers, we can define a partial order on the topologies. A topology $\mathcal{T}$ is said to be *more stable with respect to attacker* $\mathcal{A}$ than $\mathcal{S}$, if

$$a_{\mathcal{A}}^{\mathcal{T}}[i] \leq a_{\mathcal{A}}^{\mathcal{S}}[i] \quad \text{for } i \geq 1.$$

We denoted this relation by $\mathcal{T} \leq_{\mathcal{A}} \mathcal{S}$. A topology $\mathcal{T}$ is said to be *optimally* $\mathcal{A}$-stable if $\mathcal{T} \leq_{\mathcal{A}} \mathcal{S}$ for every topology $\mathcal{S}$.

It is neither clear, whether optimally $\mathcal{A}$-stable topologies exist for an arbitrary attacker $\mathcal{A}$, nor how they look like, if

they exist. Furthermore, if $\mathcal{T}$ is optimally $\mathcal{A}$-stable, then there might exist another attacker $\mathcal{A}'$, such that the damage caused by it is significantly higher than that caused by $\mathcal{A}$. Hence, stability against attacks depends heavily on the attacker and the topologies.

One way to circumvent this problem—up to a certain degree—is the usage of a reference attacker, which we choose as the *optimal attacker* $\mathcal{O}$, i.e., $\mathcal{O}$ always chooses the set with i nodes causing the maximal possible damage. Hence, we obtain the following *reference damage*:

$$a^{\mathcal{T}}[i] := a^{\mathcal{T}}_{\mathcal{O}}[i] = \max\{a^{\mathcal{T}}(X) \mid X \subseteq V \text{ and } |X| = i\}.$$

Similarly, we define $f^{\mathcal{T}}[i]$ as the maximum damage caused by the failure of i nodes:

$$f^{\mathcal{T}}[i] := f^{\mathcal{T}}_{\mathcal{O}}[i] = \max\{f^{\mathcal{T}}(X) \mid X \subseteq V \text{ and } |X| = i\}.$$

This reference damage is an upper bound for the damage caused by an arbitrary attack against the topology $\mathcal{T}$. Hence, a topology is expected to be more stable against arbitrary attacks if $a^{\mathcal{T}}[i]$ is as low as possible, or more formal, if it is optimally $\mathcal{O}$-stable or *optimally stable*.

Since the number of heads in an arbitrary topology $\mathcal{T} \in \mathbf{C}(n, k, C)$ is bounded by $C \cdot k$, we have $a^{\mathcal{T}}[i] = n \cdot k$ for $i \geq C \cdot k$.

At this point, one may ask whether optimally stable topologies exist. In fact, we are going to describe a specific class of optimally stable topologies. This class allows us to deduce some properties of topologies and guidelines for their online construction, leading to a high stability.

By the following result, it is absolutely sufficient to restrict to the attack stabilities.

Lemma 1. *For every topology $\mathcal{T} \in \mathbf{C}(n, k, C)$ and $1 \leq i \leq n$, we have $f^{\mathcal{T}}[i] = a^{\mathcal{T}}[i] - k \cdot i$.*

Proof. We have

$$\begin{aligned} f^{\mathcal{T}}[i] &= \max\{a^{\mathcal{T}}(X) - k \cdot |X| \mid X \subseteq V \text{ and } |X| = i\} \\ &= \max\{a^{\mathcal{T}}(X) \mid X \subseteq V \text{ and } |X| = i\} - k \cdot i \\ &= a^{\mathcal{T}}[i] - k \cdot i. \end{aligned}$$

□

3.4 Sequential Attacks

In general, an attack strategy is nothing but a sequence $(X_1, X_2, \ldots, X_l)$ of vertex sets $X_i \subseteq V$ with $|X_i| \leq i$. If the attacker wants to achieve a specific damage r, he has to choose an X_i of his strategy with $a^{\mathcal{T}}(X_i) \geq r$. We call such a strategy *optimal*, if $a^{\mathcal{T}}(X_i) = a^{\mathcal{T}}[i]$ for $1 \leq i \leq n$.

In the following, we will concentrate on a specific type of attack strategies, the *sequential attack strategies*, in which the attacker takes down nodes one by one in a specific order $(v_1, \ldots, v_l) \in V^l$. For an attack strategy $(v_1, \ldots, v_l)$, we can define a general attack strategy $(V_1, V_2, \ldots, V_l)$ by setting $V_i = \{v_1, \ldots, v_i\}$.

In the following, all considered attack strategies are sequential. Furthermore, the nodes in sequential attack strategies will be denoted by lower case letters, e.g., $(u_1, \ldots, u_l)$ and the sets of the corresponding general attack strategies will be denoted by the corresponding upper case letters, e.g., $(U_1, \ldots, U_l)$.

Several types of attack strategies for a given topology $\mathcal{T}$ are possible.

- A *random strategy* is an arbitrary sequence of nodes, chosen randomly while the attack is conducted.
- A *greedy strategy* $(v_1, \ldots, v_n)$ is constructed by choosing the node with maximal additional damage at each step, i.e., v_1 is a node with

  $$a^{\mathcal{T}}(v_1) = \max\{a^{\mathcal{T}}(v) \mid v \in V\},$$

  and after the selection of $v_1, \ldots, v_i$, the next node v_{i+1} is chosen, such that

  $$a^{\mathcal{T}}(V_i \cup \{v_{i+1}\}) = \max\{a^{\mathcal{T}}(V_k \cup \{v\}) \mid v \in V \setminus V_i\}.$$

- An *optimal strategy* is an attack strategy $(v_1, \ldots, v_l)$ such that

  $$a^{\mathcal{T}}[i] = a^{\mathcal{T}}(V_i)$$

  for $0 \leq i \leq l$, i.e., the set V_i achieves maximum damage among all sets of at most i nodes.

While random and greedy strategies always exist, it is not clear that every topology $\mathcal{T}$ has an optimal strategy. But it is easy to see that—if it exists—every optimal strategy is greedy.

As Lemma 1 already indicates, stability against attacks and against failure are equivalent. The next result shows this equivalence regarding optimal attack strategies.

Lemma 2. *An attack strategy $(v_1, \ldots, v_l)$ on $\mathcal{T} \in \mathbf{C}(n, k, C)$ is optimally stable, if and only if $f^{\mathcal{T}}[i] = f^{\mathcal{T}}(V_i)$ for $1 \leq i \leq l$.*

Proof. Let $(v_1, \ldots, v_l)$ be optimal. Then, we have $f^{\mathcal{T}}[i] = a^{\mathcal{T}}[i] - ik = a^{\mathcal{T}}(V_i) - ik = f^{\mathcal{T}}(V_i)$. If on the other hand, $f^{\mathcal{T}}[i] = f^{\mathcal{T}}(V_i)$, then

$$a^{\mathcal{T}}[i] = f^{\mathcal{T}}[i] + ik = f^{\mathcal{T}}(V_i) + ik = a^{\mathcal{T}}(V_i).$$

□

4 Optimally Stable Streaming Topologies

In the following, we will describe topologies $\mathcal{T}$, which are optimally stable, i.e., every other topology $\mathcal{S}$ allows to achieve the same amount of damage by removal of at most the same number of nodes, as $\mathcal{T}$.

As we will see, the optimally stable topologies described in the following, correspond to the intuition, that the heads have to be the most important nodes and they have to be close to equally important. In fact, in general topologies, it is possible that an attack on nonheads, which are hubs in many stripes, might be more efficient, contradicting this intuition. The problem of finding an optimal attack is NP-complete and can only be approximated in polynomial time up to a factor of $\log(n)$, as we prove in another article [16]. Furthermore, the Greedy-Attacker is not optimal on general topologies.

4.1 The Optimal Topologies

In this section, we will describe a class of topologies in $\mathbf{T}(n, k, C)$ with $n \geq C \cdot k$, whose members are optimally stable. Before we give a set of mathematical properties, which these topologies have to satisfy, we describe a way to

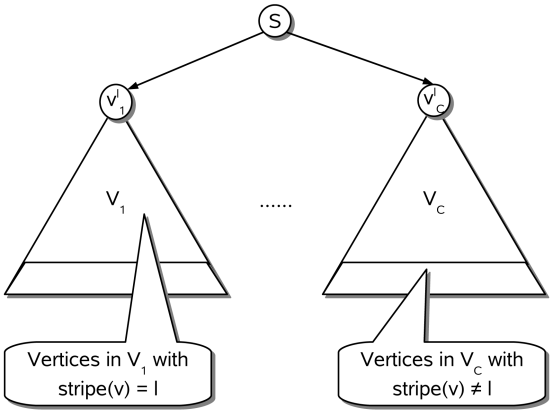

Fig. 1. Scheme of an optimal topology.

construct a family satisfying our definition. A rough scheme of this construction is shown in Fig. 1.

1. Choose a set $\mathcal{H}$ of $C \cdot k$ nodes, which will become the heads of the topology $\mathcal{T}$.
2. Split $\mathcal{H}$ into C disjoint groups $\mathcal{H}_1, \ldots, \mathcal{H}_C$, each containing exactly k members. Assume that $\mathcal{H}_h = \{v_h^1, \ldots, v_h^k\}$. The nodes $v_1^l, \ldots, v_C^l$ will become the C heads of stripe l.
3. To each node $v \in V \setminus \mathcal{H}$, assign an arbitrary number $\mathrm{stripe}(v) \in \{1, \ldots, k\}$, and set $\mathrm{stripe}(v_h^l) := l$ for $v_h^l \in \mathcal{H}$.
4. For each stripe $l = 1, \ldots, k$, do the following:
   a. Split V into C disjoint groups $V_1, \ldots, V_C$, such that
      i. for i, j, we have $\big||V_i| - |V_j|\big| \leq 1$, i.e., all groups have approximately the same size;
      ii. $\mathcal{H}_j \subseteq V_j$.
   b. For each group V_h, $h = 1, \ldots, C$, arrange the nodes in V_h in a rooted tree, such that
      i. v_h^l is the root and has at least $k-1$ successors;
      ii. $v \in V$ is a leaf if $\mathrm{stripe}(v) \neq l$.
      This step of the construction is quite vague and contains several degrees of freedom, which will be discussed later.
   c. For each group V_h, $h = 1, \ldots, C$, connect v_h^l to the server s.

This constructive procedure has several nonobvious effects.

- Every node is a leaf in at least $k-1$ stripes, i.e., every node replicates packets of at most one stripe.
- Heads do not mix, i.e., if a head v is a successor of another head v' in an arbitrary stripe, then in every other stripe, either v' is a successor of v or both, v and v' are successors of the same head.
- We have $|\mathrm{succ}_l(v_h^l)| = |\mathrm{succ}_{l'}(v_h^{l'})|$ for $l, l' = 1, \ldots, k$ and $h = 1, \ldots, C$.

Definition 3. *A topology $\mathcal{T} \in \mathbf{T}(n, k, C)$ with $n \geq C \cdot k$ is called optimal if it satisfies the following conditions:*

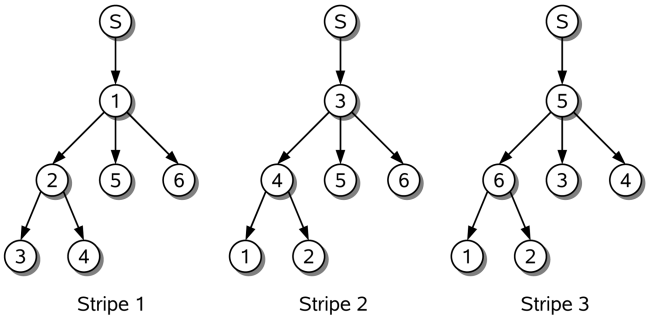

Fig. 2. A simple optimal topology for $C = 1$, $k = 3$, and $n = 6$.

1. *Every node v is a leaf in at least $k-1$ stripes.*
2. *The lth stripe, $l = 1, \ldots, k$, has exactly C heads $v_1^l, \ldots, v_C^l$.*
3. *For v_h^l, we have*
$$\left\lfloor \frac{n}{C} \right\rfloor \leq |\mathrm{succ}_l(v_h^l)| \leq \left\lceil \frac{n}{C} \right\rceil.$$
4. $|\mathrm{succ}_l(v_h^l)| = |\mathrm{succ}_{l'}(v_h^{l'})|$ *for* $l, l' = 1, \ldots, k$ *and* $h = 1, \ldots, C$.
5. *For $l, l' = 1, \ldots, k$ and $h, h' = 1, \ldots, C$, we have $v_{h'}^{l'} \in \mathrm{succ}(v_h^l)$ if and only if $h' = h$.*
6. *Every head has at least $k-1$ direct successors.*

In this context, the notion *optimal topology* is a bit misleading. As we will see these topologies are, in fact, optimally stable. But other measures are completely ignored, and the quality of our *optimal topologies* regarding these is absolutely unclear. In fact, *optimal topology* is simply a name for a class of topologies, which turn out to be optimally stable.

At this point, we do not consider whether optimal topologies can be constructed in networks with nodes of limited capacity. If every node has a sufficiently high capacity, it is quite obvious that optimal topologies can be constructed by choosing trees of depth 1 as templates for step 4b in the construction. In this case, in every stripe, the heads are chosen, and then every other node is connected directly to its corresponding head. Another way would be to chose a family of trees with I interior nodes and at least $(k-1) \cdot I$ leaves and use them as templates for step 4b of the construction (like the one in Fig. 2).

4.2 The Optimal Sequence

Now that we know our candidates for optimal topologies, we want to find an optimal attack strategy for them. As we will see the sequence $\big(a^{\mathcal{T}}[i]\big)_{1 \leq i \leq n}$ for an optimal topology, $\mathcal{T}$ is closely related to the sequence $(\delta_i)_{1 \leq i \leq C \cdot k}$ with

$$\delta_i := \begin{cases} \left\lceil \frac{n}{C} \right\rceil + (k - 2l - 1), & \text{if } h \leq n \mod C, \\ \left\lfloor \frac{n}{C} \right\rfloor + (k - 2l - 1), & \text{if } h > n \mod C, \end{cases}$$

where $i = C \cdot l + h$ with $0 \leq l < k$ and $1 \leq h \leq C$. Observe that l and h are uniquely determined by

$$l = \left\lfloor \frac{i-1}{C} \right\rfloor \quad \text{and} \quad h = i - C \cdot l.$$

As we will see, the values δ_i satisfy $\delta_i = a^{\mathcal{T}}[i] - a^{\mathcal{T}}[i-1]$ and there exists an optimal attack strategy $(v_1, \ldots, v_{C\cdot k})$ with $a^{\mathcal{T}}(V_i) = a^{\mathcal{T}}[i]$.

The representation $i = C \cdot l + h$ of an index i is caused by the way the optimal strategy is constructed. Basically, the attacker runs through the stripes and removes all heads of a stripe, before skipping to the next stripe. As a consequence, the ith node of the optimal attack strategy with $i = C \cdot l + h$ is the hth head in the $(l+1)$th stripe.

Lemma 4. *There exists an attack strategy $(v_1, \ldots, v_{C\cdot k})$ of an optimal topology $\mathcal{T} \in \mathbf{T}(n, k, C)$ with $n \geq C \cdot k$, such that*

$$a^{\mathcal{T}}(V_i) = \sum_{j=1}^{i} \delta_j \qquad \text{for } 1 \leq i \leq C \cdot k.$$

Proof. Let $v_1^1, \ldots, v_C^1, v_1^2, \ldots, v_C^2, \ldots, v_1^k, \ldots v_C^k$ be the heads of $\mathcal{T}$, such that v_h^l is the hth head in the lth stripe. Furthermore, assume that inside the lth stripe, the heads are ordered nonincreasingly, i.e., $|\mathrm{succ_l(v_1^l)}| \geq \ldots \geq |\mathrm{succ}_l(v_C^l)|$. Let $S_h^l = \mathrm{succ}_l(v_h^l)$ be the set of successors of v_h^l in stripe l.

For $i = 1$, we have $i = C \cdots 0 + 1$, and hence

$$\delta_1 = \begin{cases} \left\lceil \frac{n}{C} \right\rceil + (k-1), & \text{if } 1 \leq n \mod C, \\ \left\lfloor \frac{n}{C} \right\rfloor + (k-1), & \text{if } 1 > n \mod C. \end{cases}$$

Due to the properties of optimal topologies, we have

$$a^{\mathcal{T}}(v_1^1) = \sum_{i=1}^{k} a_i^{\mathcal{T}}(v_1^1) = \left\lceil \frac{n}{C} \right\rceil + (k-1),$$

proving the proposed equality for $i = 1$.

Now assume that $i > 1$. We have $i = C \cdot l + h$ with $0 \leq l < k$ and $1 \leq h \leq C$. Since the heads of the first l stripes are in V_i, we have $\mathrm{succ}_l(V_i) = V$, and hence, $a_j^{\mathcal{T}}(V_i) = n$ for $j \leq l$.

In the $(l+1)$th stripe, the first h heads and the heads of preceding stripes are removed. Since the heads are leaves in all but one stripe, this leads to

$$\mathrm{succ}_{l+1}(V_i) = S_1^{l+1} \cup \cdots \cup S_h^{l+1} \cup \mathcal{H}^1 \cup \cdots \cup \mathcal{H}^l.$$

Due to the properties of optimal topologies, every head of the $(l+1)$th stripe has exactly l heads of preceding stripes as successor. This implies

$$a_{l+1}^{\mathcal{T}}(V_i) = h \cdot \left\lfloor \frac{n}{C} \right\rfloor + \min(h, n \bmod C) + l \cdot (C - h).$$

In stripe $j > l+1$, only leaves (heads of preceding stripes) are removed, leading to

$$\mathrm{succ}_j(V_i) = \mathcal{H}^1 \cup \cdots \mathcal{H}^l \cup (\mathcal{H}^{l+1} \cap S_h^{l+1}),$$

and since the $\mathcal{H}^j$ are pairwise disjoint, this leads to

$$a_j^{\mathcal{T}}(V_i) = l \cdot C + jh \text{ for } j > l+1.$$

Now it is easy to see that

$$a^{\mathcal{T}}(V_i) = \sum_{j=1}^{k} a_j^{\mathcal{T}}(V_i) = \sum_{j=1}^{i} \delta_j.$$

□

At this point, we know that the sequence δ_i for $1 \leq i \leq C \cdot k$ is given by a specific strategy of an optimal topology. Our next aim is to prove that this strategy is, in fact, optimal for $\mathcal{T}$. This is done in two steps: First, we prove that for every set of nodes, there exists a set of heads with equal or less members, causing a higher failure. In the second step, we prove that the attack strategy introduced in the preceding lemma is optimal among all strategies only involving heads.

Lemma 5. *Let $\mathcal{T} \in \mathbf{T}(n, k, C)$ be an optimal topology with $n \geq C \cdot k$. For every set Y of nodes not only containing heads, there exists a set X, such that $|X| \leq |Y|$, and $f^{\mathcal{T}}(X) \geq f^{\mathcal{T}}(Y)$ and X contains less "nonheads" than Y.*

Proof. For an arbitrary head v, let Y_v be the set of all nodes in Y, which are successors of v, but no leaves.

If $v \in Y$ for every head v with nonempty set Y_v, then we can set $X = \{v \in Y \cap \mathcal{H} \mid Y_v \neq \emptyset\} \subseteq Y$ and obtain $\mathrm{succ_l(Y) \setminus Y \subseteq succ_l(X) \setminus Y \subseteq succ_l(X) \setminus X}$, implying $f_l^{\mathcal{T}}(X) \geq f_l^{\mathcal{T}}(Y)$ for $l = 1, \ldots, k$ and, hence, $f^{\mathcal{T}}(X) \geq f^{\mathcal{T}}(Y)$.

Now assume that there exists a head v of stripe l with $Y_v \neq \emptyset$ and $v \notin Y$. Set $X = (Y \cup \{v\}) \setminus Y_v$, i.e., replace Y_v by v. Obviously, we have $\mathrm{succ}_l(Y_v) \setminus Y \subseteq \mathrm{succ}_l(v) \setminus \{v\}$. Furthermore, the direct successors of v in stripe l cannot be members of $\mathrm{succ}_l(Y_v) \setminus Y_v$ since otherwise $v \in Y_v$. But they are members of $\mathrm{succ}_l(v)$ and hence $|\mathrm{succ}_l(Y_v) \setminus Y| + (k-1) \leq |\mathrm{succ}_l(v) \setminus \{v\}|$. Since the successors of all elements in $Y \cap X$ do not change, we obtain $f_l^{\mathcal{T}}(Y) + (k-1) \leq f_l^{\mathcal{T}}(X)$. In all other stripes $j \neq l$, the removal of Y_v from Y may cause an increase of $f_j^{\mathcal{T}}(Y)$, while the addition of v may decrease $f_j^{\mathcal{T}}(Y)$ by one. As a consequence, we have $f_j^{\mathcal{T}}(Y) - 1 \leq f_j^{\mathcal{T}}(X)$ for $j \neq l$. In total, this leads to $f^{\mathcal{T}}(Y) \leq f^{\mathcal{T}}(X)$.

Since X is obtained from Y by removing at least one nonhead and adding a head, the number of nonheads is reduced. □

By iterating the preceding lemma, until the number of nonheads is decreased to zero, we obtain the following result.

Corollary 6. *Let $\mathcal{T} \in \mathbf{T}(n, k, C)$ be an optimal topology with $n \geq C \cdot k$. Then, for every set Y of nodes, there exists a set X of heads, such that $|X| \leq |Y|$ and $f^{\mathcal{T}}(X) \geq f^{\mathcal{T}}(Y)$.*

Now we prove that the strategy of Lemma 4 is optimal among all strategies only involving heads.

Lemma 7. *Let $\mathcal{T} \in \mathbf{T}(n, k, C)$ be an optimal topology with $n \geq C \cdots k$ and $(v_1, \ldots, v_{C\cdot k})$ an arbitrary strategy consisting of heads only. Then, $a^{\mathcal{T}}(V_i) \leq \sum_{j=1}^{i} \delta_j$ for $1 \leq i \leq C \cdot k$.*

Proof. Let $\mathcal{H}_h := \{v_h^1, \ldots, v_h^k\}$ be a group of heads. Then, the removal of m of its members causes a total damage of $m \cdot g + m \cdot (k - m)$, where $g = |\mathrm{succ}_l(v_h^l)|$. The first summand is caused by the fact that the removal of m heads in the group affects all their successors, while the second term is caused by the fact that in the unaffected $(k - m)$ stripes, exactly m leaves fail.

Since only heads inside the same group are successors of each other, failures of heads in different groups do not affect each other. Hence, let m_h be the number failing heads in $\mathcal{H}_h$ with $m = \sum_{h=1}^{C} m_h$ and $g_h = |\mathrm{succ}(v_h^l)|$ for some l. Then, the total loss is given by

$$\sum_{h=1}^{C}(m_h \cdot g_h + (k - m_h) \cdot m_h) = mk + \sum_{h=1}^{C} m_h g_h - \sum_{h=1}^{C} m_h^2.$$

Now assume that $m_h \geq m_{h'} + 2$ for $h \neq h'$. Without loss of generality, we may assume that $h = 1$ and $h' = 2$. Then

$$\begin{aligned}
&(m_1 - 1)g_1 + (m_2 + 1)g_2 - (m_1 - 1)^2 - (m_2 + 1)^2 \\
&\quad = m_1 g_1 + m_2 g_2 - m_1^2 - m_2^2 + (g_2 - g_1) + 2(m_1 - m_2) - 2 \\
&\quad \geq m_1 g_1 + m_2 g_2 - m_1^2 - m_2^2 - 1 + 4 - 2 \\
&\quad > m_1 g_1 + m_2 g_2 - m_1^2 - m_2^2,
\end{aligned}$$

since $|g_1 - g_2| \leq 1$, due to the properties of optimal topologies. As a consequence, the total damage is maximal if the number of failures inside two groups $\mathcal{H}_h$ and $\mathcal{H}_{h'}$ differ by at most 1.

Now assume that there exist two groups $\mathcal{H}_h$ and $\mathcal{H}_{h'}$ with $m_h = m_{h'} + 1$ but $g_h = g_{h'} - 1$. Without loss of generality, we may again assume that $h = 1$ and $h' = 2$. Then

$$\begin{aligned}
&(m_1 - 1)g_1 + (m_2 + 1)g_2 - (m_1 - 1)^2 - (m_2 + 1)^2 \\
&\quad = m_1(g_1 + 1) + m_2(g_2 - 1) - m_1^2 - m_2^2 \\
&\quad = m_1 g_1 + m_2 g_2 - m_1^2 - m_2^2 + m_1 - m_2 \\
&\quad > m_1 g_1 + m_2 g_2 - m_1^2 - m_2^2,
\end{aligned}$$

and hence, the damage would be increased, if m_h and $m_{h'}$ are exchanged. Hence, for the optimal strategy, it is necessary that $m_h > m_{h'}$ implies $g_h \geq g_{h'}$. The canonical strategy satisfies this condition.

Since the total loss depends only on the m_h and g_h, this implies the proposed optimality of the canonical strategies among all attack strategies only involving heads. □

A consequence of Corollary 6 and Lemma 7 is the optimality of the attack strategy of Lemma 4 in optimal topologies.

Theorem 8. *Let $\mathcal{T} \in \mathbf{T}(n, k, C)$ be an optimal topology with $n \geq C \cdots k$. Then, $a^{\mathcal{T}}[i] = \sum_{j=1}^{i} \delta_j$.*

Proof. Let Y be an arbitrary set of nodes with $|Y| \leq k \cdot C$. Due to Lemma 5, there exists a set X of heads with $|X| \leq |Y|$ and $f^{\mathcal{T}}(X) \geq f^{\mathcal{T}}(Y)$. Now assume that Y' is a set of heads with $|Y| = |Y'|$. Then, we have

$$\begin{aligned}
\sum_{j=1}^{|Y|} \delta_j &\geq a^{\mathcal{T}}(Y') = f^{\mathcal{T}}(Y') + k \cdot |Y| \\
&\geq f^{\mathcal{T}}(X) + k \cdot |Y| \geq f^{\mathcal{T}}(Y) + k \cdot |Y| = a^{\mathcal{T}}(Y).
\end{aligned}$$

By Lemma 4, we have $a^{\mathcal{T}}[i] = \sum_{j=1}^{i} \delta_j$. □

4.3 The Optimality of Optimally Stable Topologies

At this point, we know an optimal strategy for optimal topologies and we know the maximum damages. What remains to be proven is that the optimal topologies are, in fact, among the most stable topologies in $\mathbf{T}(n, k, C)$. But before we proceed with the proof of this fact, we observe

$$\sum_{j=1}^{C \cdot l} \delta_j = l \cdot n + lC(k - l),$$

and for $i = C \cdot l + h$ with $0 \leq l < k$ and $1 \leq h \leq C$ and $n \geq C \cdot k$, we have

$$\begin{aligned}
\delta_i &\geq (k - l) + \left(\left\lfloor \frac{n}{C} \right\rfloor - l - 1\right) \\
&\geq (k - l) + (k - (k - 1) - 1) = (k - l).
\end{aligned}$$

In the following, let $\mathcal{T}$ be an arbitrary topology, such that the stripes are ordered nondecreasingly in the number of their heads, i.e., $|\mathcal{H}^1| \leq |\mathcal{H}^2| \leq \cdots \leq |\mathcal{H}^k|$. Define $H_l = \sum_{j=1}^{l} |\mathcal{H}^j|$. Let $(v_1, \ldots, v_{h_k})$ be a *redundant* strategy, i.e., a member may occur multiple times, such that $\mathcal{H}^l = \{v_{H_{l-1}+1}, \ldots, v_{H_l}\}$, i.e., the members of $\mathcal{H}^l$ form a subsequence. Furthermore, let these subsequences be ordered greedily, i.e., such that inside them, $a^{\mathcal{T}}(V_{i+1}) - a^{\mathcal{T}}(V_i)$ is nonincreasing.

Since V_{H_l} contains the heads of the first l stripes, we have $a^{\mathcal{T}}(V_{H_l}) \geq l \cdot n + H_l(k - l)$. Here, the first summand is a lower bound for the caused damage in the first l stripes and the second summand is a lower bound for the damage in the remaining $(k - l)$ stripes.

Now set i and j, such that $H_l = C \cdot i + j$ and $0 \leq i < k$ and $1 \leq j \leq C$. Since the stripes are ordered nondecreasingly by the number of their heads, and since the number of heads is limited by $C \cdot k$, we have $H_l \leq C \cdot l$ for $1 \leq l \leq k$. This implies $i < l$ and due to $k - l \leq \delta_{C \cdot l + x}$ for $1 \leq x \leq C$, we obtain

$$\begin{aligned}
a^{\mathcal{T}}(V_{H_l}) &\geq l \cdot n + H_l(k - l) \geq \sum_{j=1}^{C \cdot l} \delta_j - (C \cdot l - H_l)(k - l) \\
&\geq \sum_{j=1}^{C \cdot l} \delta_j - \sum_{j=H_l+1}^{C \cdot l} \delta_j = \sum_{j=1}^{H_l} \delta_j \geq \sum_{j=1}^{|V_{H_l}|} \delta_j.
\end{aligned}$$

Now consider the sets Y_i for $1 \leq i \leq C \cdot k$, containing the first i *distinct* nodes of the strategy, or $Y_i = H$ if $i \geq |H|$. For the same reasons as above, we have

$$a^{\mathcal{T}}(Y_{C \cdot l}) \geq l \cdot n + C \cdot l(k - l) = \sum_{j=1}^{C \cdot l} \delta_j.$$

In total, there exists a sequence of indices $1 = I_1 < I_2 < \cdots < I_{l-1} < I_l = C \cdot k$, such that either $I_j = C \cdot l$ and/or $I_j = H_l$ for some l, and

$$a^{\mathcal{T}}(Y_{I_i}) \geq \sum_{j=1}^{I_i} \delta_j.$$

For each subsequent pair of those indices, the subsequence $(v_{I_i}, \ldots, v_{I_{i+1}-1})$ of the strategy lies in H^l for some l and between $Cl' + 1$ and $C \cdot l' + C$ for some other l'. As a consequence, the δ_j inside this sequence form a nonincreasing sequence with $|\delta_{I_{j+1}-1} - \delta_{I_j}| \leq 1$. In addition, the $\alpha_j := a^{\mathcal{T}}(Y_{j+1}) - a^{\mathcal{T}}(Y_j)$ form a nonincreasing sequence. Due to Lemma 11 in the Appendix, this implies

$$a^{\mathcal{T}}(Y_i) \geq \sum_{j=1}^{i} \delta_j,$$

for $1 \leq i \leq Ck$, and hence, we obtain the following theorem.

Theorem 9. *Assume $n \geq C \cdot k$ and let $\mathcal{T}$ be an arbitrary topology in $\mathbf{T}(n, k, C)$ with $n \geq C \cdot k$. Then, there exists a strategy $(v_1, \ldots, v_{C \cdot k})$ on $\mathcal{T}$, such that for $1 \leq i \leq C \cdot k$,*

$$a^{\mathcal{T}}(X_i) \geq \sum_{j=1}^{i} \delta_j.$$

The most important consequence of this theorem is the global optimal stability of optimal topologies.

Theorem 10. *An optimal topology $\mathcal{T} \in \mathbf{T}(n, k, C)$ with $n \geq C \cdot k$ is optimally stable.*

Proof. Let $\mathcal{S}$ be an arbitrary topology in $\mathbf{T}(n, k, C)$ and $\mathcal{T}$ an optimal topology. Then, due to Theorems 9 and 8, we have $a^{\mathcal{S}}[i] \geq \sum_{j=1}^{i} \delta_j = a^{\mathcal{T}}[i]$ for $1 \leq i \leq C \cdot k$. □

5 System Design

Creating optimally stable topologies in real networks cannot easily be done. The highly dynamic behavior of joining and departing nodes in live multimedia streaming scenarios leads to the fact that a centralized approach cannot scale to large groups of participating nodes.

A distributed approach to constructing optimally stable topologies using a deterministic procedure has to fail for a number of reasons.

Global knowledge, which is needed to create optimal topologies, is impossible to gather in a distributed situation and under the circumstances of possible message loss and node failures. Even presuming reliable communication, the system would suffer from a high message overhead and the system would not be scalable to large groups. An additional drawback of distributing the needed information for an explicit construction of an optimally stable topology is the fact that it leads to knowledge about the placement of nodes, which a malicious party could use for an attack.

The heterogeneity of real nodes regarding bandwidth and processing resources, and constant variation of the availability of these resources, are further obstacles for the creation of optimally stable topologies. In the described topologies, every node needs only to be able to forward the bitrate of the received stream once. However, while some nodes may have significantly more available uplink capacity, it might happen that one or a number of nodes are located behind a bottleneck link with less capacity, and thus, are unable to fulfill these requirements.

In order to still be able to create topologies, which are close to optimally stable, an implicit, distributed approach is needed.

Following Section 4.1, it becomes apparent that optimally stable topologies have a distinct set of three properties: 1) every node forwards data in only one spanning tree; 2) the number of distinctive direct child nodes of the source and of all heads are maximized; 3) the difference of the number of successors of all heads is at most 1.

To achieve these properties, we design a distributed procedure. It creates overlay live streaming topologies in a tree-first approach based on local knowledge only. The main aims of the procedure are to keep the overall topology balanced and as low as possible, with each node forwarding in only one of the spanning trees.

Following the general idea of SplitStream, the stream is split into k distinctive stripes, k spanning trees are created for the transmission of the stream and every node aims at forwarding only data in the spanning tree of one of the stripes. Due to the fact, that in certain situations, a node might have to forward data in another stripe as well, this selection is not done rule-based. Instead, we choose a cost-based approach and define four different cost metrics, $K_1 \ldots K_4$, of all outgoing links of a node, which are combined in a total cost function K. K is calculated by every forwarding node in order to evaluate and optimize its local situation with regard to the stability of the local section, including the node itself and all of its successors, of the overall topology. Nodes may bootstrap to the system by joining at any node which is already part of the topology and they may leave and possibly even rejoin at any time. To maintain the topology and its properties, every node v calculates the total cost of the edges $e = (v, w)$ to its child nodes in the spanning tree of stripe i: $K(v, w, i) = \sum_{j=1}^{4} s_j \cdot K_j(e, i)$. Through modification of the local topology, the cost of its situation is then minimized. The defined cost metrics follow four simple objectives.

A node needs to choose the stripe which it prefers to forward. As bandwidth constraints may lead to a demand for additional nodes which can forward in one of the stripes, this selection should not be static. Hence, every node selects the stripe which it forwards to the most child nodes as its preferred stripe and assigns higher cost to all edges in the spanning trees of different stripes

$$K_1(v, i) := 1 - \frac{\text{fanout}_{T_i}(v)}{c^+(v)},$$

with $fanout_{T_i}(v)$ being the number of outgoing edges of node v in the spanning tree T_i and $c^+(v)$ being the bandwidth capacity available for outgoing stripes; hence, $c^+(v) = c(v) - 1$.

While all participants need to receive the data of all stripes, only a selection of the nodes chooses to forward the data in the spanning tree of a certain stripe. To avoid that nodes have to forward in more than one spanning tree, it is important for each spanning tree to keep bandwidth capacity available. The selection of nodes that is able to forward has to be kept in a low layer in the regarding spanning tree in order to keep the spanning trees low, and thus, to permit an easy location of the available bandwidth while optimizing and short optimization paths. Thus, edges to nodes, which have chosen to forward data in a considered stripe, are assigned low cost, whereas edges to nodes, which choose to forward data in a different stripe, are assigned a higher cost

$$K_2(v, w, i) := \begin{cases} 0, & \text{neighbor can forward in stripe i,} \\ 1, & \text{else.} \end{cases}$$

The topologies are balanced using a third cost function, which is used by a node to level the number of successors of its child nodes

$$K_3(v,w,i) := \frac{\left(\frac{\mathrm{succ}_{T_i}(v)}{\mathrm{fanout}_{T_i}(v)} - 1\right) - \mathrm{succ}_{T_i}(w)}{\left(\frac{\mathrm{succ}_{T_i}(v)}{\mathrm{fanout}_{T_i}(v)} - 1\right)}.$$

In the distributed environment, the amount of successors $\mathrm{succ}_{\mathrm{T_i}}$ of a node is lazily gathered as a reverse multicast.

In case a node has to forward data in more than one of the spanning trees, it is important that the direct dependency to each child node is kept to a minimum. The last cost metric hence evaluates the number of direct connections between a node v and each node u of its child nodes and aims at keeping them to a minimum:

$$K_4(v,w) := \frac{\mathrm{fanout}_{\mathcal{T}}(v,u)}{k},$$

where $\mathrm{fanout}_{\mathcal{T}}(v,u)$ is the number of stripes in which v directly forwards the stripe to u.

In previous work [17], we have analyzed topologies which were created using only one or a subset of the defined cost metrics. However, the results showed that all four cost metrics have to be used in order to achieve a good stability of the topologies.

To create stable topologies in real networks, we implemented a distributed algorithm, which cooperatively optimizes the topology by local modifications of the neighborhood of nodes using the defined total cost function. The algorithm is run on each node to optimize their local situation through rearranging direct links. In order to be able to achieve locally optimal topologies, each node would have to be able to optimize its complete neighborhood. However, child nodes are not aware of the situation at their parent nodes. Hence, through redirecting a succeeding child node to its own parent, oscillations could be caused, since a situation might occur in which a parent without available bandwidth keeps redirecting a node to one of it's successors, which, in turn, due to its local optimization, keeps redirecting it back to its parent. Since these oscillations have to be avoided, nodes optimize the cost of the links to their child nodes only, while the links to the parents are not optimized.

Local rearrangements of the topology are performed in the two phases (cf., Algorithm 1). In the first phase, the cost for all outgoing links are minimized following three steps (lines $6 \ldots 13$). Using the total cost function K, the link with the highest cost is determined. As the cost metrics make use of knowledge on the local situation, and a parent does not necessarily possess this information for the situation of the child nodes, the total cost cannot be minimized straightforward. The parent node instead calculates the gain of dropping the selected link to one of the other child nodes considering only K_2 and K_3, thus balancing the trees as well as possible while dropping links to nodes which volunteer to forward in the associated stripe only. The gain for dropping the link e to the alternative parent u with link $e' = (v,u)$ is calculated as $G(v,u,i) = K_3(v,u,i) - K_2(v,u,i)$. It is important that the height of the topologies stays at a minimum level and, hence, the modifications are only done if the gain through the cost minimization exceeds the threshold Θ_{pass}. This threshold cannot be selected as a constant value, but it needs to be a function of the available bandwidth of the node. A node, which has just entered the system and does not serve any other nodes, needs to be utilized as a forwarding peer in the system in order to keep the height of the trees as low as possible. However, a node which already serves other nodes and gets close to its bandwidth limitations needs to be able to pass child nodes to other nodes for service. Thus, the threshold is calculated as a function of the available bandwidth of a node.

```
   Input: v, i,
   Childs_{T_i}(v) {child nodes of node v in T_i}
 1 d ← ∅ {link to drop};
 2 a ← ∅ {alternative parent};
 3 b ← deg(v);
 4 gain ← true;
 5 i ← preferred stripe;
 6 while gain do
 7 |   gain ← false;
 8 |   d ← argmax {K(v,w,i) | w ∈ Childs_{T_i}(v)} ;
 9 |   a ← argmax {G(v,w,i) | w ∈ Childs_{T_i}(v)\{d}};
10 |   if G(e',i) ≥ Θ_pass(v) then
11 |   |   drop(d, a);
12 |   |   gain ← true;
13 |   end
14 end
15 while b < c(v) do
16 |   a ← w = rand{Childs_{T_i}};
17 |   childRequest(a,Θ_pass(v), (succ_{T_i}(v)/fanout_{T_i}(v) − 1));
18 |   b ← b + 1;
19 end
```

After analyzing the local situation and repeating the three steps of phase one until no more child nodes are dropped to alternative parents, the nodes check their bandwidth consumption. If they have bandwidth capacities available, they request successors from their child nodes in the second phase, in order to decrease the amount of levels of the topology again (lines $14 \ldots 18$). In order to avoid the topologies to oscillate, the parent node sends information about its situation to the selected child, which then is able to predetermine that it does not pass a successor, which will be dropped back immediately.

The time complexity of this algorithm for each optimization at each node v is linear in the number of its child nodes. Every node exchanges a number of messages in $\mathcal{O}(c(v) \cdot k)$. With the node's capacity $c(v)$ being limited by a constant upper bound and k being the constant amount of stripes in the system, the message complexity of the implemented protocol is in $\mathcal{O}(n)$. Each node arrival and departure may have an effect on the overall optimum, and consequently may require reorganizations of the overall topologies. To still avoid oscillations and to guarantee a fast convergence of the system as a whole, even under the most adverse conditions expected, several functional and nonfunctional means are implemented. The first encompass the optimization of outgoing links, the introduction of the threshold Θ_{pass} and the fact that node requests are only served by child nodes if their parent's state after is estimated to be

stable. The latter mainly consists of the fact that the topologies are optimized to form broad and well-balanced trees, resulting in a low height, and thus, avoiding long optimization paths.

6 Performance Comparison

In order to evaluate the stability of the optimally stable topologies and compare them with topologies of both the proposed procedure and existing ALM approaches, we compared their stability under attacks. For this purpose, we analyzed instances of the different topologies with algorithms implementing an optimal attack, based on global knowledge.

The experiments were conducted in two steps. At first, we created topologies resembling the optimal topologies using a generator. The generator constructs topologies following the approach of creating balanced spanning trees for all stripes with minimum height and disjoint sets as forwarding nodes. For comparison, we then constructed topologies with simulation models of the cost-based approach, described in Section 5, and with strategies from our own previous work, constructing DAG Topologies with BCBS [13]. The simulation models were implemented using OMNeT++/INET, a discrete event simulation framework. In each simulation, one source with the bandwidth capacity of $C = 5$ published a stream, which was partitioned into $k = 4$ distinctive stripes. The stream was subscribed by a group of nodes with a maximum bandwidth capacity of $c(v) = 3$. Each node, in consequence, was able to receive the four stripes of the stream once and additionally forwards at most eight stripes to other nodes. The user behavior, which defines the node arrivals and departures, was modeled in accordance to work of Veloso et al. [18]. Their study examines traces of existing streaming services on the Internet and the resulting models are characterized by a very high, but subsequently decreasing churn in the beginning of a streaming service (interarrival times and online times are modeled as Pareto- and lognormal distributions, respectively). Using these simulation models and the user model, we simulated nodes, which in one simulation set, ran the cost-based algorithm, and in another set, used the BCBS approach, each, until a steady state was reached. At that point snapshots of the evolved topologies, consisting of 250 nodes each, where taken for further analysis.

In the second step, we analyzed the stability of the constructed topologies toward optimal global attacks, both using a greedy global attack and a branch-and-bound solver.

It is to expect that intact topologies receive an amount of $n \cdot k$ packets, as all nodes are elements in the spanning tree T_i of all stripes. After attacking the set X of nodes, $\sum_{v \in V \setminus X} \mathrm{inc_X(v)}$ forwarding links in the spanning trees remain intact. Normalizing, we get

$$\frac{\sum_{v \in V \setminus X} \mathrm{inc}_X(v)}{n \cdot k}$$

of remaining received packets after successfully attacking the set X.

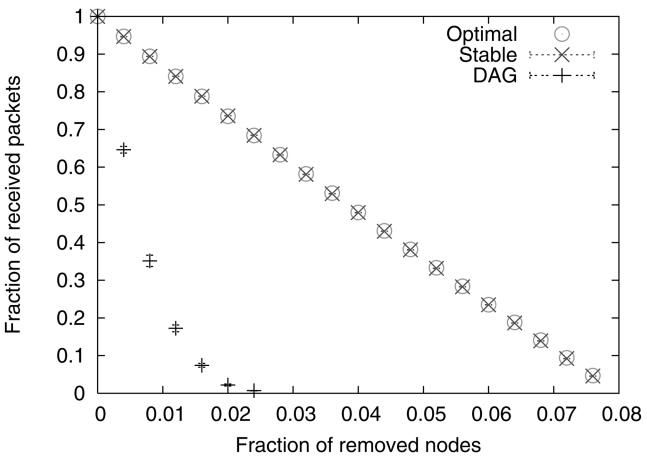

Fig. 3. Optimal attack on DAG, stable, and optimally stable topologies (with 98 percent confidence).

For optimal topologies and in case of $n = C \cdot k$, the number of intact forwarding links after attack can be calculated as $(n \cdot k) - a^{\mathcal{T}}(X)$, or normalized by $1 - \frac{a^{\mathcal{T}}(X)}{n \cdot k}$, we obtain $1 - \frac{a^{\mathcal{T}}(X)}{C \cdot k \cdot k}$. With $C = 1$, $a^{\mathcal{T}}(X)$ can be calculated as $\sum_{i=1}^{|X|} 2k - 1 - 2(i-1)$ as in the optimally stable topologies, the first removed node accounts for k stripes which are not received and $k-1$ stripes are not forwarded. The second removed node then accounts for $k-1$ stripes which are not received and $k-2$ stripes which are not forwarded, and so on. Increasing the source capacity leads to C different groups of size k. Here, at each attack, nodes are accounting for $2k - 1 - 2(i-1)$ failing stripes in their group and the resulting damage to the overall system is being reduced by $\frac{n}{C}$. After successfully attacking the $C \cdot k$ heads (all nodes, in case of $n = C \cdot k$), all nodes are separated from the source in all spanning trees and no packet is received any longer.

In case of $n > C \cdot k$, successfully attacking the $C \cdot k$ heads still leads to the complete disruption of the service. In the setup of the evaluation, in consequence, the total disruption of the service is reached after attacking $C \cdot k = 20$, which represents a subset of 8 percent of all nodes (cf., Fig. 3, *"optimal"*), in the best case.

As the number of successors of all heads are equally $\lfloor \frac{n-C}{C} \rfloor$ (differing by at most one additional node for some heads) in optimal topologies, each successful attack on a head leads to a loss of $\frac{n-C}{C}$ incoming stripes plus all incoming stripes of the attacked node. Hence, with C and k being fixed values and an increasing number n of nodes, the number of stripes which are not received with each attacked node can be approximated again as accounting for a linear loss of $\frac{n}{C}$. This, too, is well visible as a linear drop in the number of received packets in Fig. 3 (*"optimal"*).

We expected the DAG topologies that were created using BCBS to experience a higher loss of received stripes with each successfully attacked node. Thus, they are expected to being less stable than the topologies that were created using the cost-based approach, as they are neither optimized to being well balanced nor to being low. In consequence, the stable topologies, which were constructed using the cost-

Successors	12	14	16	18	20	22	28
Number of Nodes	143	72	16	1	1	1	1

Fig. 4. The distribution of the number of packets depending on nodes forwarding in more than one stripe.

Height	3	4	5	6	7
Number of Heads	233	44	32	9	2

Fig. 5. The number of heads of a specific height.

based approach, are expected to be more stable than the DAG topologies. However, as the cost-based system is implemented in a distributed procedure that optimizes the neighborhood of the nodes based on local knowledge only, it is assumed that successful attacks on nodes will still lead to a higher number of stripes not being received than in the optimally stable topologies, created using the generator with global knowledge. The stability toward an optimal attack of the topologies generated using global knowledge, finally, should meet the theoretical values.

For the last case, the optimally stable topologies, we realized that our assumptions hold true and the calculated quality drop equals the theoretical values. As Fig. 3 shows, the optimally stable topologies additionally are much more robust to attacks based on global knowledge than the DAG topologies described in [13]. However, the topologies generated using the cost-based approach were much more stable than the DAG topologies as well and even almost matched the stability of the optimally stable topologies.

Hence, worst-case node failures and perfect attacks in optimally stable topologies lead to an only slightly lower packet loss for receivers than in topologies which are optimized toward stability properties using the distributed, cost-based procedure. These, however, in turn, are much less susceptible to perfect attacks than DAG topologies.

In order to compare the topologies to other systems, sample topologies were obtained through the authors, where possible, and analyzed as well. It turned out that the topologies constructed using DagStream showed an almost identical stability as the topologies which were generated using BCBS, with the latter ones being slightly more stable toward attacks. This fact is not surprising, as the approaches are very similar, BCBS only allows for a more fine-grained splitting of the stream, as it creates topologies of spanning trees at packet level rather than at a stripe level. However, a more fine-grained partition of the stream with DagStream should lead to topologies which are as stable as the topologies created using BCBS. Both for Magellan and SplitStream, not enough topologies could be acquired which would be needed to conduct experiments leading to significant results. However, both systems create unbalanced and rather high topologies. These properties lead to the fact that while they are stable to random node failures, they cannot be stable toward optimal attacks.

In order to further evaluate the topologies of the cost-based approach, we analyzed to what extent they complied to the three properties required in Section 5: 1) each node forwarding in only one stripe, 2) the number of distinctive successors of the heads being maximized, and 3) the number of successors of all heads differing by 1 at maximum.

Regarding the number of stripes in which each node is forwarding, it became apparent that an average of 6 percent of the nodes in all simulations where actually internal nodes in two, and thus in more than one stripes, with an overall minimum of 4 percent and an overall maximum of 10 percent of the nodes in any simulation. In total, we found 235 nodes in the 16 topologies, which forwarded in more than one stripe, with a minimum of nine and a maximum of 24. The average total number of successors $a^{\mathcal{T}}(v) = \sum_{i=1}^{k} |\mathrm{succ}_i(c)|$ of these nodes was 12. The precise numbers can be found in Fig. 4. Even though these nodes forward more than one stripe, the experiments indicate that the low number of packets depending on them does not cause the topologies to be less stable. In fact, we conjecture that the condition for optimally stable topologies that a node forwards only in one stripe can be relaxed.

All heads had the maximum number of direct successors in all topologies of all simulations. The number of successors of each head, finally, should be balanced to $\lfloor \frac{n-C}{C} \rfloor = \lfloor \frac{n}{C} - 1 \rfloor$ or 49 in the conducted simulations. In all simulations, a fraction of 5.6 percent of all heads had a deviating number of successors. However, in these cases, they had either 50 or 48 and, thus, only one successor more or less than the average. While this small deviation could be observed in some cases, the numbers of successors of heads were exactly balanced in 18 percent of the simulations. The main requirement, that the successors of the heads are well balanced and no succeeding node has a higher number of successors than any of the heads could thus be confirmed.

To make the results easier comparable to previous work, we additionally measured the vertex connectivity, i.e., the number of disjoint paths between a node and the source of the stream. In the total of all simulations, the measured fraction of nodes with less than the maximum of k node disjoint paths to the source was 0.4 percent. Each of these nodes received two stripes through one common node and thus had a number of $k-1$ node disjoint paths to the source. While in the worst case (one simulation run), the fraction of these nodes with a vertex connectivity of $k-1$ rose to 1.2 percent of the simulated group, this decrease of the vertex connectivity did not happen at all in 37 percent of the simulations.

Finally, we examined the height of the resulting trees to obtain a simple measure for latency. To obtain a detailed picture, we examined the height of the heads instead of the height of the root. The number of heads of a specific height is given in Fig. 5. As we see, about 73 percent of the heads had a depth of three, while 14 percent had a height of four, and 10 percent a height of five. This shows the tendency to quite flat topologies in the generated stable topologies, as well as our class of optimally stable topologies, as the condition that each node forwards only one stripe requires the stripes to have a high number of leaves (those nodes forwarding other stripes) compared to the interior nodes, and hence a low height.

With respect to the convergence of the cost-based approach, we could verify that it does not suffer from extensive rearrangements of the topologies, due to oscillations or long optimization paths. Nodes were forwarded to other parent nodes two times on average for each spanning

tree, with a total maximum of $3 \cdot k$ movements over all subscribed stripes. While these numbers may sound surprisingly low, they are actually expected, as the created topologies are quite shallow (cf., Fig. 5) and optimization paths in consequence short. The maximum number of nodes that were indirectly affected by the arrival or departure of other nodes totalled to the number of stripes. This result is not surprising and shows that our measures against oscillating topologies, in order to stabilize the system quickly, are effective. Additionally, it supports our expectations that the distributed algorithm converges quickly and the overall topology is only slightly affected by the churn of nodes. Hence, reorganizations to a larger extent are only expected in the case that major changes occur. However, as the convergence of the algorithm was not in the focus of our research, the measurements taken are not concise, a task that we are planning for future studies.

7 Conclusion and Future Work

In this paper, we introduced a graph theoretic model for application layer multicast overlays. We subsequently described different types of strategies to attack topologies, including the definition of an optimal attack strategy. Based on the knowledge of optimal strategies, we were able to describe overlay streaming topologies, which are optimally stable toward perfect attacks and gave proof of their optimality. Furthermore, using this class of optimally stable topologies, we were then able to present a distributed procedure, which creates stable topologies through cost-based optimizations of the neighborhood of each node based on local knowledge only. The different topologies were evaluated in a simulation study, which revealed that the topologies created through the distributed procedure were much more stable than topologies from previous approaches, and that they were actually close to optimally stable against worst-case node failures and perfect attacks.

As real-world scenarios are commonly assumed to be characterized by a high churn of arriving and departing nodes and a high rate of random node failures, we plan to extend our research toward analyzing, how stable our new topologies are in such an environment. We expect them to also be more stable in this case. However, they might not be more stable to a very high extent than topologies created using SplitStream or DagStream/BCBS, when subject to a random removal of nodes. If the topologies created with competing approaches developed scale free characteristics with respect to a power-law-distributed amount of successors, they could even be more stable to random node failure. This would be consequence of the fact that in such topologies, the chances of an important node with many successors to fail are very low, compared to the chances of one of the many nodes with only very few successors to fail. However, first results with a random attack strategy show that the gap between the stability of our topologies and the topologies of previous approaches shrinks, with our new topologies still being more stable under all circumstances. This result most probably stems from the fact that the topologies of existing approaches do not display very strong scale free characteristics. Further studies are planned to analyze the convergence behavior of the cost-based approach in adverse environments with high churn. The preliminary results are very promising, as they show that the measures taken in our approach are effective. Additionally, since the simulation study we conducted takes a user model into account that introduces a very high but decreasing churn at the start of the simulation, and which has been developed following real-world measurements of multimedia services on the Internet, we are confident that it will perform well in these studies.

By applying some basic rules in our procedure in step 4b of the construction (cf., Section 4.1), it should be possible to create the topologies such that the number of successors is distributed—at least approximately—by a Power Law. In this way, the constructed topologies will be close to optimally stable and may show a high resilience against random failures.

Appendix

The following Lemma is used in the proof of Theorem 9.

Lemma 11. *Let $(x_i)_{1\le i\le k}$ and $(y_i)_{1\le i\le l}$ be two nonincreasing sequences of natural numbers, and x_0, y_0 two natural numbers, such that*

- $x_0 \ge y_0$,
- $\sum_{i=0}^{k} x_i \ge \sum_{i=0}^{l} y_l = Y$, *and*
- $y_i \in \{\lfloor (Y-y_0)/k \rfloor, \lceil (Y-y_0)/k \rceil\}$.

Then, for $0 \le h \le \min(k,l)$, we have

$$\sum_{i=0}^{k-h} x_i \ge \sum_{i=0}^{l-h} y_i.$$

Proof. If $k \le l$, we have $0 \le h \le \min(k,l) = k$. For $h = k$, the proposed inequality follows from $x_0 \ge y_0$. For $h < k$, we obtain by induction

$$\sum_{i=0}^{k-h-1} x_i \ge \sum_{i=0}^{l-h-1} y_i.$$

Now assume that

$$\sum_{i=0}^{k-h} x_i < \sum_{i=0}^{l-h} y_i.$$

This implies $x_{k-h} < y_{l-h}$, and hence, due to the third condition, $x_{k-h} \le \lfloor (Y-y_0)/k \rfloor$, implying

$$\sum_{i=0}^{k} x_i \le \sum_{i=0}^{k-h} x_i + h\left\lfloor \frac{Y-y_0}{k} \right\rfloor < \sum_{i=0}^{l-h} y_i + \sum_{i=l-h+1}^{l} y_i = Y,$$

contradicting the second condition.

If $k > l$, then define $\hat{x}_0 := x_0 + \cdots + x_{k-l-1}$ and $\hat{x}_i := x_{i+(k-l)}$ for $1 \le i \le l$. Then, the proposed inequality follows from the case $k \le l$. □

References

[1] Y.H. Chu, S.G. Rao, S. Seshan, and H. Zhang, "A Case for End System Multicast," *IEEE J. Selected Areas in Comm.*, vol. 20, no. 8, pp. 1456-1471, Oct. 2002.

[2] V. Pai, K. Kumar, K. Tamilmani, V. Sambamurthy, and A. Mohr, "Chainsaw: Eliminating Trees from Overlay Multicast," *Proc. Fourth Int'l Workshop Peer-to-Peer Systems*, 2005.

[3] X. Hei, Y. Liu, and K. Ross, "IPTV over P2P Streaming Networks: The Mesh-Pull Approach," *IEEE Comm. Magazine,* pp. 86-92, 2008.
[4] M. Zhang, Q. Zhang, L. Sun, and S. Yang, "Understanding the Power of Pull-Based Streaming Protocol: Can We Do Better?" *IEEE J. Selected Areas in Comm.,* vol. 25, pp. 1678-1694, 2007.
[5] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application Layer Multicast," *ACM Computer Comm. Rev.,* pp. 205-217, 2002.
[6] D. Carra, R.L. Cigno, and E.W. Biersack, "Graph-Based Analysis of Mesh Overlay Streaming Systems," *IEEE J. Selected Areas in Comm.,* vol. 25, pp. 1667-1677, 2007.
[7] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan, "Resilient Multicast Using Overlays," *ACM SIGMETRICS Performance Evaluation Rev.,* pp. 102-113, 2003.
[8] S. Birrer, D. Lu, F. Bustamante, Y. Qiao, and P. Dinda, "FatNemo: Building a Resilient Multi-Source Multicast Fattree," *Proc. Ninth Int'l Workshop Web Content Caching and Distribution,* pp. 182-196, 2004.
[9] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-Bandwidth Multicast in Cooperative Environments," *Proc. 19th ACM Symp. Operating Systems Principles,* pp. 298-313, 2003.
[10] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron, "Scribe: A Large-Scale and Decentralized Application-Level Multicast Infrastructure," *IEEE J. Selected Areas in Comm.,* vol. 20, no. 8, pp. 1489-1499, 2002.
[11] A. Rowstron and P. Druschel, "Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems," *Proc. Int'l Federation for Information Processing (IFIP)/ACM Int'l Conf. Distributed Systems Platforms,* pp. 329-350, Nov. 2001.
[12] J. Liang and K. Nahrstedt, "DagStream: Locality Aware and Failure Resilient Peer-to-Peer Streaming," *Multimedia Computing and Networking,* vol. 6071, pp. 224-238, 2006.
[13] T. Strufe, G. Schäfer, and A. Chang, "BCBS: An Efficient Load Balancing Strategy for Cooperative Overlay Live-Streaming," *Proc. IEEE Int'l Congress on Comm. (ICC),* pp. 304-309, 2006.
[14] T. Strufe, J. Wildhagen, and G. Schäfer, "Towards the Construction of Attack Resistant and Efficient Overlay Streaming Topologies," *Electronic Notes in Theoretical Computer Science,* vol. ENTCS1692, no. 179C, pp. 111-121, 2007.
[15] S. Birrer and F.E. Bustamante, "Magellan: Performance-Based, Cooperative Multicast," *Proc. 10th Int'l Workshop Web Content Caching and Distribution,* pp. 133-143, 2005.
[16] S. Grau, M. Brinkmeier, M. Fischer, and G. Schaefer, "On the Complexity and Approximability of Perfect Attacks on Multiple-Tree p2p Streaming Topologies," *IEEE Trans. Dependable and Secure Computing,* 2008.
[17] T. Strufe, "A Peer-to-Peer-Based Approach for the Transmission of Live Multimedia Streams (German: "Ein Peer-to-Peer-Basierter Ansatz für Die Live-ÜBertragung Multimedialer Daten")," PhD dissertation, TU Ilmenau, 2007.
[18] E. Veloso, V. Almeida, W. Meira, A. Bestavros, and S. Jin, "A Hierarchical Characterization of a Live Streaming Media Workload," *Proc. ACM Internet Measurement Workshop,* pp. 117-130, 2002.
[19] J. Wildhagen, "A Signalling Procedure for the Resource Location and Topology Control in Cooperative Multimedia Streaming Systems (German: "Signalisierung für die Ressourcenlokalisierung und den Topologieaufbau in Kooperativen Multimedia-Streaming-Systemen")," Master's thesis, TU Ilmenau, 2006.

Michael Brinkmeier received the graduate degree in mathematics in 1996, the Doctorate degree, for his work on operads, algebraic topology, and homotopy theory, from the University of Osnabrück, Germany, in 2000, and the Habilitation degree from the Technische Universität Ilmenau in 2008. He is currently working at the Institute of Theoretical Computer Science, Technische Universtität Ilmenau. His research interests lie in the areas of algorithms on graphs and hypergraphs and network analysis and design. He focuses on the detection of cohesive groups and connectivity issues of (hyper) graphs, and the construction of resilient network topologies. He is a member of the ACM and the German Gesellschaft fuer Informatik (Computer Science Society).

Günter Schäfer received the Diploma and PhD degrees in computer science from the University of Karlsruhe, Germany, in 1994 and 1998, respectively. Between February 1999 and July 2000, he took a post at the Ecole Nationale Superieure des Telecommunications, Paris, France, where he focused on network security and access network performance of third-generation mobile communication networks. Between August 2000 and March 2005, he worked at the Technical University of Berlin, Germany, in the areas of network security and advanced mobile communication architectures and services. Since April 2005, he has been a full professor of telecommunications/computer networking at the University of Ilmenau, Germany. His main subject areas are network security, protection of communication infrastructures, as well as communication protocols and architectures. He is a member of the ACM, the IEEE, and the German Gesellschaft fuer Informatik (Computer Science Society).

Thorsten Strufe received the PhD degree, which deals with the construction of both efficient and robust overlay streaming topologies, from Technische Universität Ilmenau, in 2007. He subsequently held a postdoc at EURECOM, working mainly on security of online social networks and currently is a professor of peer-to-peer networks at the University of Darmstadt since April 2009. His research interests lie in the areas of security in decentralized distributed systems ("P2P," opportunistic networks) and network analysis, with an emphasis on studying social networks and new possibilities to establish trust and reputation in these environments. He focuses on nonfunctional properties of networks and is especially interested in the construction of resilient and secure systems. He is a member of the IEEE and the German Gesellschaft für Informatik.