Distributed Graph-based Topology Adaptation using Motif Signatures*

Michael Stein† Karsten Weihe‡ Augustin Wilberg† Roland Kluge§
Julian M. Klomp† Mathias Schnee‡ Lin Wang† Max Mühlhäuser†

Abstract

A motif is a small graph pattern, and a motif signature counts the occurrences of selected motifs in a network. The motif signature of a real-world network is an important characteristic because it is closely related to a variety of semantic and functional aspects. In recent years, motif analysis has been successfully applied for adapting topologies of communication networks: The motif signatures of very good networks (e.g., in terms of load balancing) are determined a priori to derive a target motif signature. Then, a given network is adapted in iterative steps, subject to side constraints and in a distributed way, such that its motif signature approximates the target motif signature.

In this paper, we formalize this adaptation problem and show that it is $\mathcal{NP}$-hard. We present LoMbA, a generic approach for motif-based graph adaptation: All types of networks, all selections of motifs, and all types of consistency-maintaining constraints can be incorporated.

To evaluate LoMbA, we conduct a simulation study based on several scenarios of topology adaptation from the domain of communication networks. We consider topology control in wireless ad-hoc networks, balancing of video streaming trees, and load balancing of peer-to-peer overlays. In each considered application scenario, the simulation results are remarkably good, although the implementation was not tuned toward these scenarios.

1 Introduction

Motifs, graph patterns of small size, are an essential building block of complex networks [16]: In the last decade, a plethora of research papers, e.g., [5, 8, 20], have analyzed motif distributions of real-world graphs, showing that the motif signature, which counts the occurrences of selected motifs, is closely related to a variety of semantic and functional aspects.

Recently, first efforts have been made in the domain of communication networks to use motifs not only for the analysis of graphs but also for their adaptation. The motif signatures of very good networks are determined a priori, and a given network is adapted in iterative steps, subject to side constraints and in a distributed way, such that its motif signature approximates a target motif signature. In [7, 9, 21], algorithms have been developed for specific application scenarios. In the domain of communication networks, a graph models the network topology. By adapting this graph at runtime, application-specific optimization metrics can be addressed [24], e.g., energy consumption or load balancing. The existing algorithms for motif-based adaptation are limited to specific motif signatures, systems or graph constraints. Among others, this limitation makes it difficult to transfer the idea of motif-based adaptation to new scenarios within or beyond the domain of communication networks.

Our Contributions We summarize the contributions we make in this paper as follows: *i*) We formalize the problem of adapting a graph toward a given target motif signature, and prove that the problem is $\mathcal{NP}$-hard. *ii*) We propose a generic heuristic named LoMbA (**Lo**cal **M**otif-**b**ased **A**daptation) to solve the problem. LoMbA is executed in a distributed way on each node of the graph, and conducts two major steps. First, the algorithm selects a search space of possible graph modifications. Second, LoMbA chooses graph modifications from this search space to adapt the graph toward the target motif signature while ensuring consistency-maintaining graph constraints. *iii*) We conduct an extensive simulation study that is based on three application scenarios: topology control in wireless ad-hoc networks, balancing of video streaming trees, and load balancing of peer-to-peer overlays. The results demonstrate that LoMbA is able to adapt the graph toward the target signature for each application scenario.

Because of its generality, LoMbA fosters the use of motif-based graph adaptation in novel future use cases:

- *Rapid prototyping*: LoMbA allows for rapid prototyping of motif-based graph adaptations for new application scenarios within or beyond the domain of communication networks. The developer solely specifies a consistency-maintaining constraint and target motif signatures. Suitable target signatures

*This work has been funded by the German Research Foundation (DFG) as part of projects A1 and B2 within the Collaborative Research Center (CRC) 1053 – MAKI.

†Telecooperation Group, TU Darmstadt, Darmstadt, Germany. {stein, wang, max}@tk.tu-darmstadt.de.

‡Algorithms Group, TU Darmstadt, Darmstadt, Germany. {weihe, schnee}@cs.tu-darmstadt.de.

§Real-Time Systems Lab, TU Darmstadt, Darmstadt, Germany. roland.kluge@es.tu-darmstadt.de.

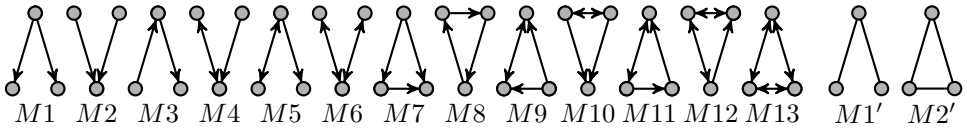


Figure 1: All directed ($M1$-$M13$) and undirected ($M1'$, $M2'$) motifs with 3 nodes

might even be learned automatically, e.g., by using genetic programming [4].

- *Dynamic signature adjustment*: LoMbA enables adjusting the target signature at runtime to react to changing optimization goals in highly dynamic environments.
- *Topology transitions*: LoMbA allows for conducting topology transitions, i.e., switching between different topologies efficiently [25]. Thereby, not only the target signature but also the consistency-maintaining constraint is replaced at runtime.

Roadmap The rest of this paper is structured as follows. Section 2 reviews the related work. Section 3 formalizes the considered adaptation problem. In Section 4, we prove that the problem is $\mathcal{NP}$-hard. Section 5 introduces the LoMbA algorithm. Our simulation study is presented in Section 6, followed by the discussion of the results in Section 7. Section 8 concludes the paper.

2 Related Work

This section discusses related work on motif analysis, network topology adaptation in general, and motif-based topology adaptation in particular.

Motif Analysis Motif analysis has first been investigated in computational biology [23] and has since been applied to a variety of network types in biology and biochemistry [20]. The underlying insight is that biological and biochemical dynamics are statistically related to the occurrence of small functional blocks, which have specific graph structures. This insight is well captured by motif signatures. The idea of functional blocks applies in domains beyond biology and biochemistry. For example, the authors of [15, 16] compare networks from different domains (e.g., biology, electrical engineering) and find that the motif signatures from different domains may serve as a fingerprint of the respective domain. Among others, further studies analyze the motif signatures of languages [2], social networks [5], reputation networks [33], and citation networks [8].

Network Topology Adaptation The topology is an essential element of any communication system. We distinguish two types of topologies: physical topologies and logical topologies. An example of a physical topology is a wireless ad-hoc network [17], where edges indicate that the incident nodes are within each other's transmission range. An example of a logical topology is a peer-to-peer overlay [27], where edges give the communication pattern between the nodes. To optimize the performance of the communication network, both physical and logical topologies may need to be adapted at runtime [24, 25]. This is achieved by modifying the graph's edge set while maintaining crucial application-specific graph constraints, e.g., regarding the increase of routing path lengths [26].

Motif-based Topology Adaptation A key observation of motif analysis is that the motif signature of a graph reflects the distribution of occurrence of functional blocks. The authors of [7, 9, 21] build upon this observation to optimize communication network topologies. For this purpose, they attack specific cases of the algorithmic problem considered in this paper.

Krumov et al. [7] adapt a video streaming tree toward a given target signature based on the motifs $M1$ and $M3$ in Figure 1. The target signature was extracted from a balanced tree. Adapting the graph toward this target signature constructs a balanced tree in a distributed way. Using a set of two graph operations that transform a rooted tree into another rooted tree, the algorithm is limited to the specific constraint of rooted tree adaptation.

In [9], Krumov et al. use the same methodology and propose two algorithms to balance the load in the peer-to-peer overlays CAN [18] and Kademlia [14]. Both algorithms ensure corresponding graph constraints by exploiting characteristics of the optimized peer-to-peer overlay. The algorithm optimizing CAN is integrated into the overlay's join procedure: When a new node joins the topology during the construction process, the algorithms decides for possible join locations whether joining there has a positive impact on achieving the target motif signature. As soon as the overlay has found a suitable location, the algorithm interrupts the join procedure. Then, the new node joins the network at this location. In a similar way, the optimization algorithm for Kademlia intercepts the procedure of adding a new

incoming connection to a node. In the case that the connection has no positive impact on the motif signature of the node, the connection attempt is rejected. Then, the overlay is enforced to search for an alternative connection. Due to their passive characteristic, both algorithms may only be used at construction time or in scenarios where the system on its own is able to propose suitable adaptations at runtime.

Schweizer et al. [21] propose kTC, a topology control algorithm for wireless ad-hoc networks. The main goal of topology control is to remove edges from a topology to allow nodes to reduce their transmission range [31]. Basically, kTC breaks triangles in undirected topologies. This corresponds to adapting a graph toward a motif signature that does not contain the motif $M2'$ in Figure 1. kTC is limited to this specific motif signature. Moreover, kTC is limited to wireless application scenarios because the algorithm only ensures the constraint that the output topology is a connected subgraph of the input topology.

The discussed existing algorithms [7, 9, 21] are limited to specific motif signatures, systems or graph constraints. In contrast, we provide a generic algorithm for the adaptation of graphs toward given motif signatures, thus facilitating motif-based rapid prototyping, dynamic signature adjustment and topology transitions.

3 Basic Definitions and Problem Specification

This section provides basic definitions and formalizes the considered adaptation problem.

Motifs A *motif* is a connected undirected or a weakly connected directed graph. Typically, but not exclusively, motifs are very small (3 to 4 nodes) [1, 16]. For a graph $G(V, E)$ and a motif M, an *occurrence* of M is a node-induced subgraph of G isomorphic to M [32]. For a given finite set $\mathcal{M} = \{M_1, \ldots, M_\ell\}$ of pairwise non-isomorphic motifs, the *motif signature* $\mathrm{s}(G)$ of G is a real-valued vector of size ℓ where the i-th entry is the relative frequency of M_i in G, that is, the number of occurrences of M_i divided by the total number of occurrences of all motifs in $\mathcal{M}$. We define the *distance* $\Delta(x, y)$ of motif signatures x and y of size ℓ as the Euclidean distance between x and y.

Communication Networks In this paper, we consider the application domain of communication networks. A *topology* represents the graph structure of a communication network whose nodes represent devices, and whose directed/undirected edges represent unilateral/bilateral communication channels. Nodes typically operate in a distributed way (e.g., in peer-to-peer overlays [27]). It is beneficial to restrict the view of each node to its local neighborhood, e.g., for scalability and robustness [29]. For a graph $G(V, E)$, a node $v \in V$, and a hop number $k \in \mathbb{N}$, the *k-neighborhood* $N(G, v, k)$ of v in G is the subgraph of G induced by all nodes reachable from v via at most k hops, where directed as well as undirected edges may be traversed in both directions.

Consistency-Maintaining Functions Let f be a Boolean function that receives a graph as input. A graph G *fulfills* f if $f(G)$ is true. Let the hop number k be given, and let f_{local} be a Boolean function that receives two graphs $N(V, E)$ and $N'(V, E')$, and a node $v \in V$ as input. N must be a node-induced subgraph of G, and all nodes in N must be reachable from v within at most k hops. Then, f_{local} is *consistency-maintaining* with respect to the constraint f if for any G, N, N', and v as defined above the following holds. If G fulfills f, and $f_{local}(N, N', v)$ is true, then the graph G' that results from replacing N in G with N' fulfills f as well; more formally:

$$\begin{aligned} & f_{local} \text{ is consistency-maintaining} \\ :\Leftrightarrow \quad & f(G) \wedge f_{local}(N, N', v) \Rightarrow f\left((G \setminus N) \cup N'\right). \end{aligned}$$

The following examples illustrate this concept:

- *Connectivity*: If for a connected graph G, $N(G, v, k)$ is replaced with a connected graph N', then the resulting graph G' remains connected.
- *Tree structure*: If a subgraph $N(G, v, k)$ of a rooted tree G is replaced with a tree N' with the same local root (i.e., the same node has in-degree 0 in $N(G, v, k)$ and N'), then the resulting graph G' remains a rooted tree.
- *Subgraph property*: If $N(G, v, k)$ is replaced with $N' \subseteq N(G, v, k)$, then the resulting graph G' is a subgraph of G.

Problem Specification
Prerequisites: *i*) A set $\mathcal{M} = \{M_1, \ldots, M_\ell\}$ of motifs; *ii*) a hop number $k \in \mathbb{N}$ that is greater than or equal to the number of nodes in any M_i; *iii*) a graph constraint f and a corresponding consistency-maintaining adaptation constraint f_{local}.
Input: *i*) A target signature $t \in \mathbb{R}^\ell$, *ii*) a graph $G(V, E)$ that fulfills f.
Output: A graph $G'(V, E')$ that fulfills f.
Objective: Minimize the distance between $\mathrm{s}(G')$ and t, i.e., $\min \Delta(\mathrm{s}(G'), t)$.
Requirements on algorithms:

- The algorithm runs in a distributed way with one process on each node of G.

Algorithm 1 LOCAL MOTIF-BASED ADAPTATION (LOMBA)

Input: k-neighborhood $N(G, v, k)$, node $v \in V$, target signature $t \in \mathbb{R}^\ell$

1: $N_{sample} \leftarrow$ GETSAMPLEGRAPH$(N, v, k, max_nodes, target_size)$
2:
3: Set<Iterator> $its \leftarrow$ {REMOVEIT(N_{sample}, v), ADDIT(N_{sample}, v), MOVEIT(N_{sample}, v)}
4:
5: $EOI \leftarrow$ GETEDGEOPERATIONINDICATOR(N_{sample}, t) // Equation (5.1)
6: List<Iterator> $itOrder \leftarrow$ GETITERATORORDER$(its, EOI, eoi_threshold)$
7:
8: **for** it : $itOrder$ **do**
9: **if** TRYITERATOR$(N, N_{sample}, v, t, it)$ **then** // Algorithm 2
10: **break** // G has been adapted successfully

- The process on a node $v \in V$ has only knowledge about its k-neighborhood, as defined above.
- A basic step of the process on a node $v \in V$ is to replace the k-neighborhood $N(G, v, k)$ by a graph N' on the same nodes where $f_{local}(N(G, v, k), N', v)$ is true.

4 Complexity Analysis

In this section, we analyze the complexity of the aforementioned problem. We first consider the following inherent decision problem and show that

THEOREM 4.1. *Given a target motif signature t, a consistency-maintaining function f, and a graph G, deciding whether there exists a graph G' such that the target motif signature is perfectly matched is $\mathcal{NP}$-complete.*

Proof. The proof is conducted using the following reduction. The decision problem to be reduced is *partition into triangles* [6]. The input is an undirected graph $\tilde{G}(\tilde{V}, \tilde{E})$. Without loss of generality, we assume that $\tilde{G}$ is connected and that $|V|$ is divisible by three. The problem is to decide whether or not V partitions into disjoint triples $\{v_1, v_2, v_3\}$ such that $\{v_1, v_2\}$, $\{v_2, v_3\}$, $\{v_1, v_3\} \in \tilde{E}$. As the motif set $\mathcal{M}$, we take all undirected motifs on 3 nodes. There are only the two motifs shown in Figure 1 in the undirected case: paths of length two ($M1'$) and triangles ($M2'$). The target signature puts full weight on triangles and zero weight on the other motif. Function f accepts a graph $G(V, E)$ if and only if $V = \tilde{V}$, $E \subseteq \tilde{E}$, and each node in V is incident to at least one edge in E. As motifs are connected, the perfect matches of this motif signature are exactly the partitions of $\tilde{G}$ into triangles, which completes the reduction as well as the proof.

The above theorem directly leads to the result that the graph adaptation problem we defined is $\mathcal{NP}$-hard. In addition, we make the following remarks:

- The function f is quite natural in general, and in particular for the considered domain of communication networks. So, in contrast to many other $\mathcal{NP}$-completeness results, this one is not based on "pathological" instances far from reality.
- The partition problem to be reduced is known to be $\mathcal{NP}$-complete for many other patterns besides triangles. Moreover, there is a variety of $\mathcal{NP}$-completeness results for restricted classes of graphs. Therefore, our motif signature approximation problem is $\mathcal{NP}$-hard for a broad range of cases. We conjecture it is $\mathcal{NP}$-hard for all but a few simple special cases.
- The reduction does not make use of the requirements on algorithms stated in Section 1, i.e., the problem remains $\mathcal{NP}$-hard even if no requirements are imposed at all. We may even restrict the problem to the case $G := \tilde{G}$.

5 Algorithm

We propose LoMbA (**Lo**cal **M**otif-**b**ased **A**daptation), a generic heuristic for the problem stated in Section 3. The algorithm is based on the following observation. If each node $v \in V$ adapts its k-neighborhood $N(G, v, k)$ toward t, the graph G adapts globally toward t. Each node $v \in V$ periodically executes Algorithm 1. The algorithm consists of two major steps: The *search space selection* step enumerates a reasonable subset of modifications of $N(G, v, k)$ (see Section 5.1). The *operation selection step* selects a consistency-maintaining operation that adapts $N(G, v, k)$ toward t (see Section 5.2).

5.1 Search Space Selection The search space selection step determines iterators over possible modifications of G. Even for the limited k-neighborhood $N(G, v, k)$ of a node $v \in V$, it is infeasible to iterate over all possible modifications of $N(G, v, k)$. Thus, it is

Algorithm 2 TryIterator

Input: k-neighborhood $N(G, v, k)$, sample graph $N_{sample} \subseteq N$, node $v \in V$, target signature $t \in \mathbb{R}^\ell$, operation iterator it

1: $foundValid \leftarrow$ false // true after a consistency-maintaining operation is found
2: $currentDistance \leftarrow \Delta(\mathrm{s}(N_{sample}), t)$
3: $max\_steps \leftarrow \infty, done\_steps \leftarrow 0$
4: **while** it.hasNext() $\wedge$ $done\_steps < max\_steps$ **do**
5: $op \leftarrow it$.next() // candidate operation for modifying G
6: $done\_steps \leftarrow done\_steps + 1$
7: $N' \leftarrow$ CreateCandidate(N, op) // does not modify N or G
8: **if** $f_{local}(N, N', v)$ **then** // checks if operation is consistency-maintaining
9: $foundValid \leftarrow$ true
10: $N'_{sample} \leftarrow$ CreateCandidate(N_{sample}, op)
11: **if** $\Delta(\mathrm{s}(N'_{sample}), t) < currentDistance$ **then**
12: $G \leftarrow (G \setminus N) \cup N', N \leftarrow N'$ // as improvement was found, modify G
13: **if** $max\_steps = \infty$ **then** // limit the number of subsequent steps
14: $max\_steps \leftarrow done\_steps$
15: $done\_steps \leftarrow 0, currentDistance \leftarrow \Delta(\mathrm{s}(N_{sample}), t)$
16: **return** $foundValid$

important to keep the search space as small as possible. For this purpose, the algorithm operates on a sample subgraph of $N(G, v, k)$, sticks to a limited set of graph operations, and uses a heuristic to determine a reasonable order for the usage of these operations.

Sample Local View Even a local view of only a few hops k may contain hundreds of nodes and edges. To limit the search space of modifications, if the number of nodes in the local view is larger than a given threshold (max_nodes), we compute a subgraph of $N(G, v, k)$ to build the optimization upon (Algorithm 1, line 1). The computed subgraph shall be a representative sample of the original graph while having bounded size in terms of nodes and edges. To construct such a subgraph, we use a method inspired by random walk graph sampling [11]. Starting from node $v \in V$ that executes LoMbA, we conduct multiple random walks over k hops each until $target\_size$ distinct nodes have been traversed. Directed as well as undirected edges may be traversed in both directions. Then, we compute the graph $N_{sample} \subseteq N(G, v, k)$ that is induced by the set of traversed nodes. As $target\_size$ is a configurable constant, the size of N_{sample} is bounded by a constant as well. The constant $target\_size$ may be chosen based on the computational resources available on the target platform.

Graph Operations Each node $v \in V$ restricts itself to three graph operations (line 3):

- *Remove-edge operation*: v removes an incident edge.
- *Add-edge operation*: v adds a not yet existing edge to a 2-hop neighbor $u \in V$.
- *Move-edge operation*: v removes an edge and adds an edge. For the selection of removable edges, v follows the same rule as for the remove-edge operation, but also considers edges incident to 1-hop neighbors of v. For the selection of addable edges, v follows the same rule as for the add-edge operation.

It has been shown in [24] that these graph operations are sufficient to describe most topology adaptations in networking applications. Nonetheless, more complex, application-specific operations could easily be added to LoMbA.

Iterator Order To further reduce the search space, each node $v \in V$ analyzes only a subset of all possible graph operations introduced before. For this purpose, we use a simple heuristic (lines 5-6).

Initially, v decides which graph operation will be most beneficial to achieve the target signature t. For this purpose, v computes the *edge operation indicator* (EOI):

$$EOI = \frac{\sum_{i=1}^{\ell}(t_i - \mathrm{s}_i(N)) \cdot |E(M_i)|}{\ell} \tag{5.1}$$

In Equation (5.1), t_i denotes the relative frequency of motif M_i in the target signature, $\mathrm{s}_i(N)$ gives the relative number of occurrences of motif M_i in the input graph N, and $|E(M_i)|$ gives the number of edges contained in

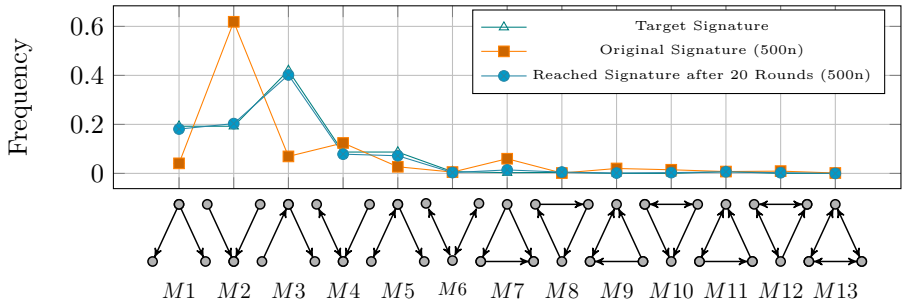


Figure 2: Target signature, original signature, and signature reached by LoMbA after 20 rounds (Scenario 3)

motif M_i. If EOI is considerably greater than 0, the average weighted amount of edges in the target signature is greater than in $N(G, v, k)$. This indicates that edges should be added to $N(G, v, k)$. Consequently, LoMbA prefers an iterator that traverses all possible add-edge operations (AddIt) based on N_{sample}. Analogously, if EOI is considerably smaller than 0, v decides to remove edges from $N(G, v, k)$ (RemoveIt). If EOI is close to 0 ($|EOI| \leq eoi\_threshold$), LoMbA prefers the move-edge iterator (MoveIt). The threshold for selecting the move-edge operation ($eoi\_threshold$) should be chosen very small. The rationale behind this is that the number of possible move-edge operations is considerably larger than the number of add-edge or remove-edge operations. This makes the move-edge iterator more expensive in terms of runtime than the other iterators.

Next, v calls the TryIterator procedure to conduct suitable graph operations based on the selected iterator (lines 8-10). This procedure is introduced in detail in the next subsection. If none of the operations are consistency-maintaining, TryIterator is called again for the other iterators to extend the search space.

5.2 Operation Selection Based on the given iterator, TryIterator (Algorithm 2) performs consistency-maintaining graph operations that adapt $N(G, v, k)$ toward the target signature. To assess a graph operation, $v \in V$ determines the candidate graph N' that would result from applying the operation to $N(G, v, k)$ (Algorithm 2, line 7). If N' violates f_{local}, the operation is discarded. Otherwise, v compares the Euclidean distance to the target signature for the motif signature of N_{sample} as well as the one of N'_{sample} (line 11). If N'_{sample} is closer to the target signature than N_{sample}, then v adapts G by replacing N with N' (line 12).

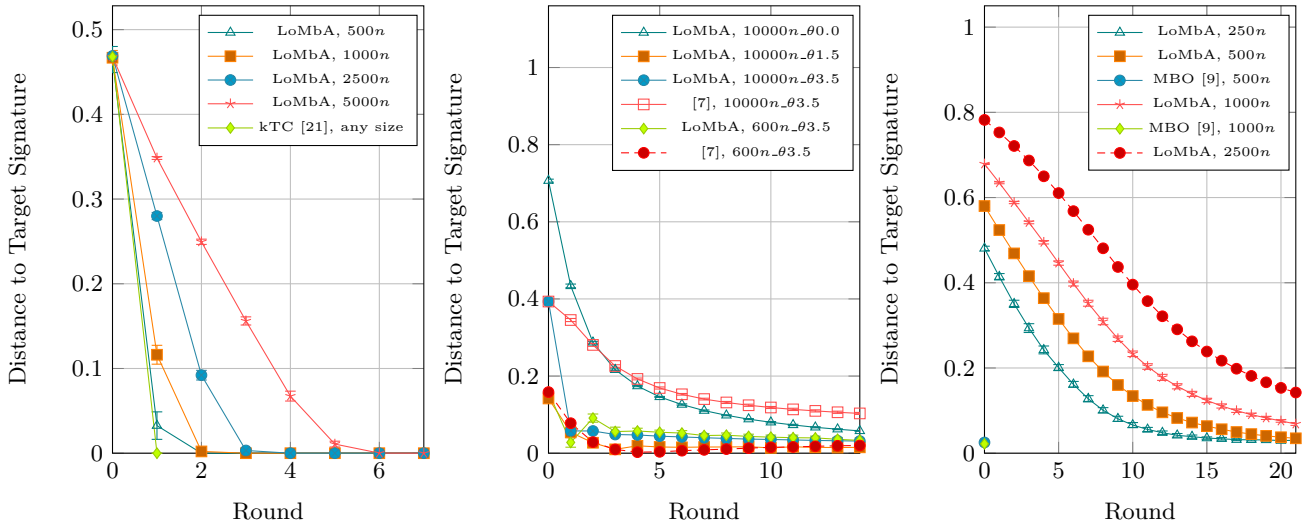
The operation assessment refers to the sample subgraph N_{sample} and not to the full local view N because of the high computational complexity of counting motifs [32]. In contrast, the consistency-maintaining constraint f_{local} is checked based on N to guarantee constraint fulfillment. If a suitable operation is found, it is likely that the iterator provides additional suitable operations. Thus, having walked $done\_steps$ steps for finding the first operation, v repeatedly conducts another $done\_steps$ steps to assess further operations (lines 13-15). If f_{local} is violated by all the operations, TryIterator returns false. As described in Section 5.1, LoMbA will then extend the search space with additional iterators. If TryIterator finds at least one consistency-maintaining operation, the procedure returns true.

6 Simulation Study

This section introduces the considered application scenarios and the set-up of our simulation study.

6.1 Scenarios To be applicable in practice, the presented generic algorithm must be able to handle a large range of graph classes and motif signatures. We consider three different application scenarios to provide a representative evaluation.

Scenario 1—Topology Control in Wireless Ad-hoc Networks The first scenario is inspired by the topology control algorithm kTC [21]. Conducting topology control in wireless networks has beneficial implications, e.g., enabling transmission range reductions [31]. We use unit disk graphs (UDGs) [10] to model wireless networks. In a UDG, the nodes are deployed in a plane and have circular transmission ranges. Two nodes are connected by an undirected edge if and only if they are within each other's range. We exported UDGs from the network simulator PeerfactSim.KOM [28]. We placed the wireless nodes uniformly at random on a squared plane with a side length of 2000 meters, using the simulator's default transmission range settings. We considered only connected UDGs, and varied the number of nodes between 500 and 5000. The basic idea of kTC is to break triangles in the topology. Two undirected



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

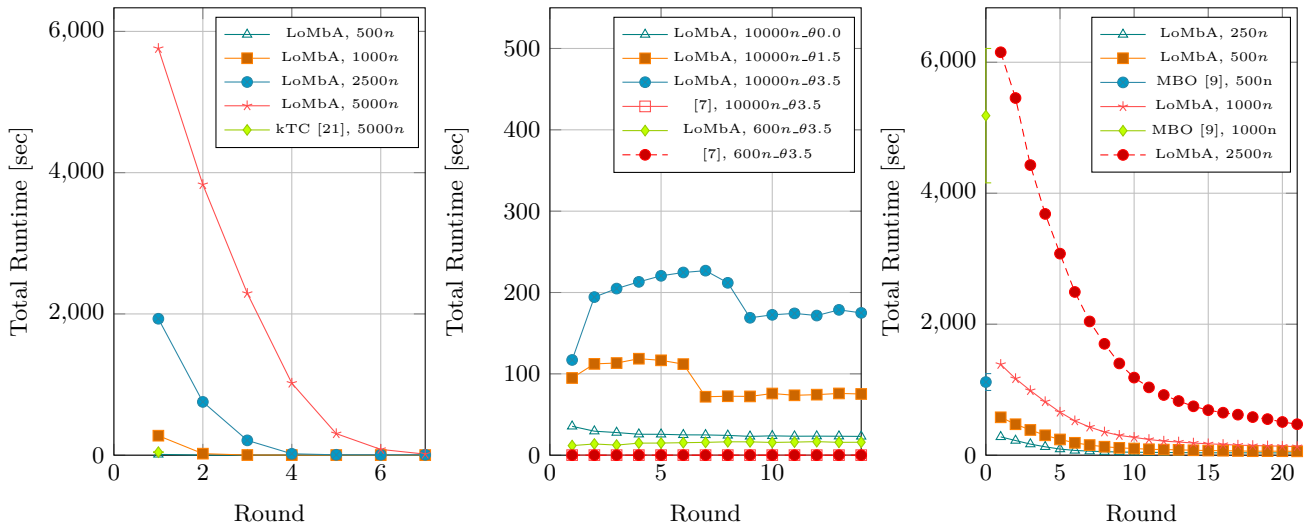
Figure 3: Distance to the target signature over the number of rounds for Scenarios 1-3

motifs with 3 nodes $\mathcal{M} = \{M1', M2'\}$ (Figure 1) exist. The target signature $t = (1.0; 0.0)$ represents all topologies that contain no triangles. In this scenario, $f_{local}(N, N', v)$ is true if N' is a connected subgraph of N. $f_{local}(N, N', v)$ is consistency-maintaining with respect to the constraint f, which states that only nodes within transmission range are adjacent in the graph. We compare our algorithm with kTC.

Scenario 2—Balancing Video Streaming Trees Krumov et al. [7] balance video streaming trees by adapting a directed rooted tree toward a motif signature with directed motifs with 3 nodes. Only the motifs $M1$ and $M3$ (Figure 1) exist in such trees. These motifs occur in a ratio $\theta = \mathrm{s}(G, M_1)/\mathrm{s}(G, M_3)$ in optimally balanced trees, where θ is based on the intended number of successors per node. We compare LoMbA with the algorithm by Krumov [7] for $\theta = 3.5$, which reflects optimally balanced trees aiming at 8 successors per node. Moreover, we varied θ for our algorithm to generate additional target motif signatures. We used PeerfactSim.KOM [28] to run the video streaming system Transit [34] and exported video streaming trees with 600 nodes. To conduct an evaluation of LoMbA for even larger graphs, we additionally used random rooted trees with 10000 nodes as input topologies. These trees were generated by sequentially appending each node at a random parent node. We defined the constraint $f_{local}(N, N', v)$ such that N' must be a rooted tree with the same root node as in N. This constraint is consistency-maintaining: It guarantees that G' is a rooted tree provided that G is a rooted tree.

Scenario 3—Load Balancing in Peer-to-Peer Overlays The load in peer-to-peer overlays such as Kademlia [14] is often unevenly distributed among the peers. Krumov et al. [9] show that the load in Kademlia can be distributed in a fair way by adapting the topology toward a certain target signature. This target signature contains 10 out of the 13 directed motifs with 3 nodes and is shown in Figure 2. Kademlia nodes possess randomly chosen identifiers, and connect with selected other nodes based on the so-called XOR metric, which is defined on the identifier space. We extended PeerfactSim.KOM [28] by a Kademlia graph generator to provide corresponding input graphs. In a first step, the graph generator adds the specified number of nodes to an initially empty graph. Then, the generator iterates over all nodes and approximates a Kademlia topology by adding suitable edges to E according to the XOR metric. We varied the graph size from 250 to 2500 nodes. The adaptation constraint $f_{local}(N, N', v)$ ensures that Kademlia's structure based on the XOR metric is preserved. We compare LoMbA with the original algorithm, called MBO [9], which intercepts the procedure of adding edges to the overlay. For this purpose, we implemented MBO as an extension of the above graph generator.

6.2 Implementation and Set-up In practice, all nodes in a communication network run asynchronously and trigger themselves periodically to adapt their local neighborhood [24]. We simulate this behavior by conducting adaptation rounds. In each round, each node is triggered exactly once and in random order. In a



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 4: Total runtime of algorithm execution including all nodes for Scenarios 1-3

real-world deployment, nodes might be triggered at the same point in time. To avoid concurrent adaptation of the same network region, our algorithm can be executed in conjunction with a locking mechanism as the one presented in [19]. We implemented LoMbA and the triggering mechanism in the Java programming language. Moreover, we implemented the ESU algorithm [32] to iterate over all subgraphs of a given size, and the VF2 algorithm [3] to check if graphs are isomorphic. Building upon this, we determined the motif signature of a graph by iterating over its subgraphs of the motif size, and comparing the subgraphs to the given set of motifs.

The computations were conducted on a 64-bit server with 26 cores (2.5GHz), 32GB of RAM, Ubuntu 14.04 and Java 8 Server VM. For each application scenario, we configured LoMbA in the same way ($k = 2$, $max\_nodes = 50$, $target\_size = 30$, $eoi\_threshold = 0.01$) and repeated the computation for 20 different input graphs.

7 Results and Discussion

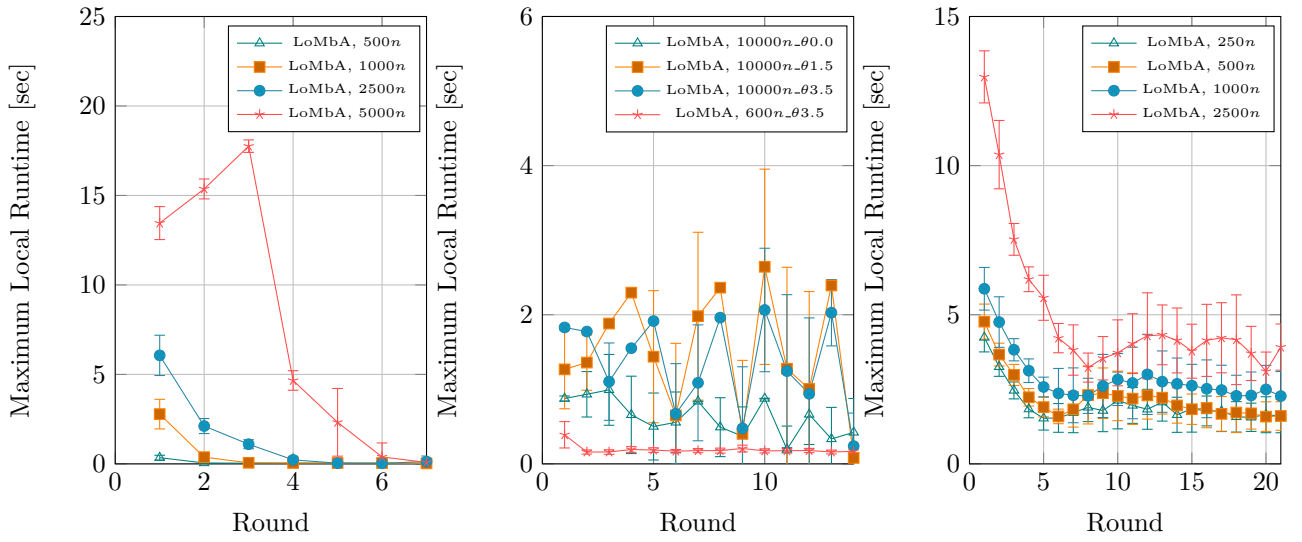
This section presents and discusses the results of our simulation study. As defined in Section 1, the objective is to minimize the distance from the motif signature of the input graph G to a target signature t while ensuring a given graph constraint f. As LoMbA fulfills f by conducting only consistency-maintaining operations, this section focuses on the aspect of adapting G toward t. Section 7.1 shows that our generic algorithm achieves this objective for each considered application scenario. Moreover, Section 7.2 demonstrates the feasibility of generic motif-based adaptation in terms of computational overhead. Section 7.3 investigates further metrics that are of particular interest in the domain of communication networks. Finally, Section 7.4 discusses possible termination schemes for LoMbA.

All plots in this paper give averaged results over 20 different input topologies, and error bars denote the standard deviation. The error bars are hardly visible in some plots because the standard deviation is very small.

7.1 Achieving the Target Motif Signature

Figure 3 shows the distance to the target signature over the number of rounds for each scenario.

Scenario 1 For the wireless ad-hoc network scenario, LoMbA perfectly achieves the given target signature (see Figure 3a). The initial graphs have a distance to t of slightly below 0.5. Based on the edge operation indicator (see Section 5.1), LoMbA always decides correctly that edges should be removed rather than added because only the motif with the fewest possible edges ($M1'$) exists in t. Removing an edge from a triangle in the local view of a node reduces the distance from this node to t. As no new triangles ($M2'$) may be formed by removing an edge from G, the distance to t never increases for any node in the network. Consequently, the graph eventually has the motif signature t. As soon as the target signature has been achieved, the graph is not modified anymore. The required number of rounds for achieving t depends on the density of the initial graph, which increases with the number of nodes because of a fixed plane size. The reason for this is that the target



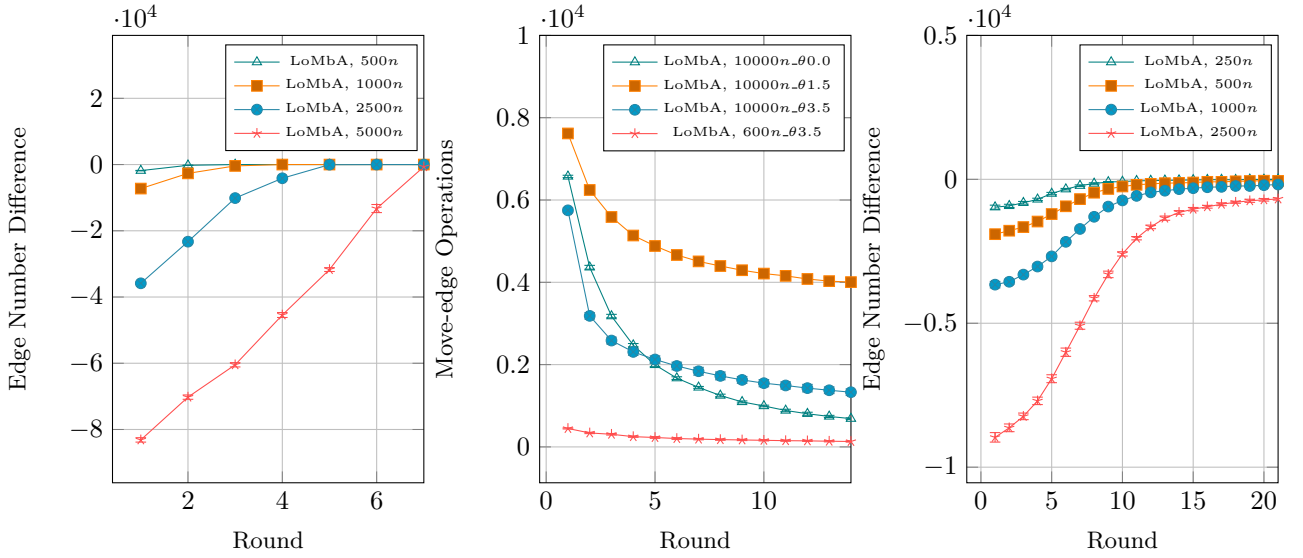
(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 5: Maximum local runtime observed for any node

topology is a sparse graph where all triangles have been removed. Consequently, the higher the density of the initial graph, the more edges need to be removed. The application-specific algorithm kTC [21] achieves the target signature in exactly one round, independent of the network density. This is achieved by iterating over all triangles found in the topology and removing the longest edge from each triangle. It is important to note that kTC has three unfavorable properties: *i*) The signature is fixed. *ii*) kTC is unable to consider constraints apart from the subgraph constraint, which is tailored to the wireless application domain. Triangle breaking has also been used in other domains (e.g., P2P topology mismatching [13]) with different topological constraints. kTC may not be used in these domains. *iii*) As kTC is unable to add edges, a re-execution of the algorithm based on the full topology is required whenever the input topology changes. LoMbA addresses these three issues with acceptable overhead introduced.

Scenario 2 For the video streaming scenario, LoMbA reduces the distance to the target signature to less than 0.06 for each configuration (see Figure 3b). Achieving crucial distance reductions in the first round, LoMbA even outperforms the specialized tree adaptation algorithm [7] for both types of graphs (random trees of size 10000 and Transit [34] trees with 600 nodes). In the subsequent rounds, LoMbA still achieves slight improvements. The signature $\theta = 3.5$ reflects a balanced tree of low height. To evaluate LoMbA for further signatures, we also investigated smaller values of θ for the topology with 10000 nodes. The extreme case is $\theta = 0$, which reflects a chain of nodes. Even this case is tackled successfully by LoMbA in less than 15 rounds. In general, as adding or removing single edges destroys the tree property of the input graph, LoMbA relies solely on the move-edge operation to adapt the graph.

Scenario 3 The goal in the third scenario is to adapt Kademlia topologies toward the target signature t depicted in Figure 2. As shown in Figure 3c, the distances from the input topologies to t range from 0.48 to 0.78. For each number of nodes, LoMbA achieves continuous distance reductions. Figure 2 shows the target signature and, exemplarily for the topology with 500 nodes, the original signature and the reached signature. LoMbA adapts the frequency of each individual motif almost perfectly toward t. A direct comparison of LoMbA with the original algorithm MBO [9] is impossible because MBO adapts the network already at construction time, while LoMbA adapts the network continuously in a round-based way. However, we compare the topology constructed by MBO with the topology adapted by LoMbA after a reasonable number of rounds. For this purpose, we included MBO into our Kademlia graph model. For 500 and 2500 nodes, the application-specific algorithm MBO is superior in terms of the achieved motif signature: The output graphs of MBO have a distance of 0.025 and 0.022 for 500 nodes and 1000 nodes, respectively. For these graph sizes, the distances of LoMbA's output graphs after 21 rounds range from 0.035 to 0.069, still reflecting distance re-



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 6: For Scenarios 1 and 3, relative number of edges added/removed from round to round; for Scenario 2, number of performed move-edge operations

ductions of about 94% and 90%, respectively. However, it is important to state that MBO may only be used in combination with Kademlia because MBO is inherently tailored for Kademlia's procedure of adding new connections. While the general idea of forbidding unsuitable connections might be generalized to other systems, MBO inherently relies on the system to propose possible topology adaptations due to its passive characteristic.

7.2 Runtime Figure 4 gives the total runtime (i.e., the sum of all nodes' individual execution times) for each application scenario. First of all, we observe that the runtime is feasible in all considered cases. In most configurations, the runtime for individual nodes is far below 1 second in average. As decision-making of each node $v \in V$ depends only on its k-neighborhood $N(G, v, k)$, the runtime of v is independent of the network size $|V|$ and instead depends on the density of $N(G, v, k)$. A practical consequence is that the runtime reduces from round to round for Scenarios 1 and 3. The target motif signatures of these scenarios facilitate sparse networks. Thus, getting closer to the target signature results in executing LoMbA on a graph with lower density. The best example is Scenario 1, where the total runtime is close to 0 in the final rounds before reaching the target signature.

The highest runtime of one round for Scenario 1 was 5758 seconds (5000 nodes), which reflects slightly more than one second for individual nodes in average. The reason for this comparably large runtime is that the corresponding input graphs have a very high average node degree of about 63. For Scenario 3, the highest round time was 6153 seconds (2500 nodes), again being caused by the high density of the initial topology. In contrast, Scenario 2 has small runtimes because of a rather low density of the input graphs.

The comparison with the related work demonstrates again the price one has to pay for generality. The lower runtimes of the specialized algorithms kTC [21] and the tree adaptation algorithm [7] can be explained by the fact that these algorithms only have to analyze the modifications specific to their motif signature and graph constraints. MBO [9] and LoMbA have a similar computational overhead. However, given that topology adaptations are executed only occasionally based on the degree of network dynamic (e.g., two times per hour in case of sensor networks [22]), the runtime overhead is almost negligible for all considered algorithms.

7.3 Further Metrics This subsection discusses further metrics that are of particular interest in the domain of communication networks.

Load Distribution Figure 4 shows that LoMbA introduces negligible overhead in terms of runtime for the network as a whole. Figure 5 additionally shows that not only the average overhead but also the maximum overhead observed for any individual node is feasible. This demonstrates that LoMbA is able to distribute the load among the nodes in a fair way.

Performed Graph Operations Figure 6 shows which operations are used by LoMbA to adapt the network topology. In particular, this helps to understand why the runtime of LoMbA varies from round to round. In Scenario 1, the graph density drops fundamentally, leading to a corresponding drop in the runtime. In Scenario 2, only the move-edge operation may be used to adapt the graph. The average density of the graph remains constant, thus leading to only small changes in the runtime per round. In Scenario 3, all graph operations are used to adapt the graph. However, the corresponding target motif signature also facilitates a sparse graph structure.

7.4 Termination As shown in Figure 3 and discussed in the previous subsections, LoMbA adapts the input graph G toward the target motif signature t in a small number of rounds. It would be desirable to stop the algorithm once convergence to t has been achieved. LoMbA runs in a distributed way, and each node may only compute the motif signature of its k-neighborhood. Therefore, it is unfortunately impossible to stop the algorithm automatically when the target signature has been achieved globally for G. However, a node may decide locally to stop triggering of LoMbA as soon as the target signature has been achieved for its k-neighborhood. Moreover, the variance of convergence time is very small for a specific scenario and configuration (note that the standard deviations in Figure 3 are hardly visible). Thus, based on experience of earlier executions, it would also be possible to stop the algorithm after a predefined number of rounds. Finally, a node may also decide to stop the algorithm as soon as the distance improvement compared to the previous round is below a given threshold.

8 Conclusion

In this paper, we proposed the generic algorithm LoMbA, which adapts graphs in a distributed way toward arbitrary target motif signatures while ensuring given graph constraints. We proved that this problem is $\mathcal{NP}$-hard. The simulation study shows that our algorithm tackles the problem remarkably well: In each of the three different application scenarios from the domain of communication networks, LoMbA achieves a close adaptation toward the target signature in a small number of rounds.

Thanks to its generality, LoMbA facilitates the use of motif-based graph adaptation in novel future use cases, such as motif-based rapid prototyping, dynamic signature adjustment and topology transitions. In addition, LoMbA might be useful in application domains beyond communication networks. For example, motif-based graph adaptation may be applied to optimize complex business processes [30], where nodes and edges represent activities and their dependencies, respectively.

In future work, we will optimize SmartGrid data aggregation topologies [12]. In this context, the generality of our approach allows for optimizing the aggregation topology dynamically either for high user privacy or low network load based on the current requirements. Moreover, we will switch between different SmartGrid topologies at runtime to modify the aggregation scheme.

References

[1] Uri Alon, *Network motifs: Theory and experimental approaches*, Nature Reviews Genetics, 8 (2007), pp. 450–461.

[2] Chris Biemann, Stefanie Roos, and Karsten Weihe, *Quantifying semantics using complex network analysis*, in Proceedings of the International Conference on Computational Linguistics (COLING), 2012, pp. 263–278.

[3] Luigi P. Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento, *A (sub)graph isomorphism algorithm for matching large graphs*, Pattern Analysis and Machine Intelligence, 26 (2004), pp. 1367–1372.

[4] Alexander Frömmgen, Robert Rehner, Max Lehn, and Alejandro Buchmann, *Fossa: Learning ECA rules for adaptive distributed systems*, in IEEE International Conference on Autonomic Computing (ICAC), 2015, pp. 207–210.

[5] Krzysztof Juszczyszyn, Przemysław Kazienko, and Katarzyna Musiał, *Local topology of social network based on motif analysis*, in Knowledge-Based Intelligent Information and Engineering Systems, Ignac Lovrek, Robert J. Howlett, and Lakhmi C. Jain, eds., vol. 5178 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 97–105.

[6] Richard M. Karp, *Reducibility among combinatorial problems*, in Complexity of Computer Computations, Raymond E. Miller, James W. Thatcher, and Jean D. Bohlinger, eds., The IBM Research Symposia Series, Springer US, 1972, pp. 85–103.

[7] Lachezar Krumov, Adriana Andreeva, and Thorsten Strufe, *Resilient peer-to-peer live-streaming using motifs*, in Proceedings of the International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010, pp. 1–8.

[8] L. Krumov, C. Fretter, M. Müller-Hannemann, K. Weihe, and M.-T. Hütt, *Motifs in co-authorship networks and their relation to the impact of scientific publications*, The European Physical Journal B, 84 (2011), pp. 535–540.

[9] Lachezar Krumov, Immanuel Schweizer, Dirk Bradler, and Thorsten Strufe, *Leveraging network motifs for the adaptation of structured peer-to-peer-networks*, in Proceedings of the Global Communications Conference (GLOBECOM), 2010, pp. 1–5.

[10] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger, *Ad-hoc networks beyond unit disk graphs*, in Proceedings of the Workshop on Foundations of Mobile Computing (FOMC), 2003, pp. 69–78.

[11] Jure Leskovec and Christos Faloutsos, *Sampling from large graphs*, in Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2006, pp. 1–8.

[12] Fengjun Li, Bo Luo, and Peng Liu, *Secure information aggregation for smart grids using homomorphic encryption*, in IEEE International Conference on Smart Grid Communications (SmartGridComm), 2010, pp. 327–332.

[13] Yunhao Liu, *A two-hop solution to solving topology mismatch*, IEEE Transactions on Parallel and Distributed Systems, 19 (2008), pp. 1591–1600.

[14] Petar Maymounkov and David Mazières, *Kademlia: A peer-to-peer information system based on the XOR metric*, in Proceedings of the International Workshop on Peer-to-Peer Systems (IPTPS), 2002, pp. 53–65.

[15] Ron Milo, Shalev Itzkovitz, Nadav Kashtan, Reuven Levitt, Shai Shen-Orr, Inbal Ayzenshtat, Michal Sheffer, and Uri Alon, *Superfamilies of evolved and designed networks*, Science, 303 (2004), pp. 1538–1542.

[16] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, *Network motifs: Simple building blocks of complex networks*, Science, 298 (2002), pp. 824–827.

[17] Charles E. Perkins, *Ad Hoc Networking*, Addison-Wesley Professional, 2008.

[18] Sylvia Ratnasamy, Paul Francis, Mark Handley, Scott Shenker, and Richard Karp, *A scalable content-addressable network*, in Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM), 2001, pp. 161–172.

[19] Shansi Ren, Lei Guo, Song Jiang, and Xiaodong Zhang, *SAT-Match: A self-adaptive topology matching method to achieve low lookup latency in structured p2p overlay networks*, in Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS), 2004, pp. 1–9.

[20] F. Schreiber and H. Schwöbbermeyer, *Motifs in biological networks*, in Statistical and Evolutionary Analysis of Biological Network Data, Michael P. H. Stumpf and Carsten Wiuf, eds., Imperial College Press, 2010, pp. 45–64.

[21] Immanuel Schweizer, Michael Wagner, Dirk Bradler, Max Mühlhäuser, and Thorsten Strufe, *kTC – robust and adaptive wireless ad-hoc topology control*, in Proceedings of the International Conference on Computer Communication Networks (ICCCN), 2012, pp. 1–9.

[22] Immanuel Schweizer, Ralf Zimmermann, Michael Stein, and Max Mühlhäuser, *a-kTC: Integrating topology control into the stack*, in Proceedings of the IEEE Conference on Local Computer Networks (LCN), 2015, pp. 414–417.

[23] Shai S. Shen-Orr, Ron Milo, Shmoolik Mangan, and Uri Alon, *Network motifs in the transcriptional regulation network of escherichia coli*, Nature Genetics, 31 (2002), pp. 64–68.

[24] Michael Stein, Alexander Frömmgen, Roland Kluge, Frank Löffler, Andy Schürr, Alejandro Buchmann, and Max Mühlhäuser, *TARL: Modeling topology adaptations for networking applications*, in Proceedings of the International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), 2016, pp. 57–63.

[25] Michael Stein, Roland Kluge, Dario Mirizzi, Stefan Wilk, Andy Schürr, and Max Mühlhäuser, *Transitions on multiple layers for scalable, energy-efficient and robust wireless video streaming*, in IEEE International Conference on Pervasive Computing and Communication (PerCom), 2016, pp. 1–3.

[26] Michael Stein, Géza Kulcsár, Immanuel Schweizer, Gergely Varró, Andy Schürr, and Max Mühlhäuser, *Topology control with application constraints*, in Proceedings of the Conference on Local Computer Networks (LCN), 2015, pp. 229–232.

[27] Ralf Steinmetz and Klaus Wehrle, eds., *Peer-to-Peer Systems and Applications*, vol. 3485 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2005.

[28] D. Stingl, C. Gross, J. Rückert, L. Nobach, A Kovacevic, and R. Steinmetz, *Peerfact-Sim.KOM: A simulation framework for peer-to-peer systems*, in Proceedings of the IEEE International Conference on High Performance Computing and Simulation (HPCS), 2011, pp. 577–584.

[29] Jukka Suomela, *Survey of local algorithms*, ACM Computing Surveys, 45 (2013), pp. 24:1–24:40.

[30] Wil M. P. van der Aalst, *Business process management: A comprehensive survey*, ISRN Software Engineering, 2013 (2013), p. 37.

[31] Yu Wang, *Topology control for wireless sensor networks*, in Wireless Sensor Networks and Applications, Yingshu Li, MyT. Thai, and Weili Wu, eds., Signals and Communication Technology, Springer US, 2008, pp. 113–147.

[32] Sebastian Wernicke, *Efficient detection of network motifs*, Transactions on Computational Biology and Bioinformatics, 3 (2006), pp. 347–59.

[33] Matthias Wichtlhuber, Sebastian Bücker, Roland Kluge, Mahdi Mousavi, and David Hausheer, *Of strategies and structures: Motif-based fingerprinting analysis of online reputation networks*, in Proceedings of the Conference on Local Computer Networks (LCN), 2016, pp. 1–8.

[34] M. Wichtlhuber, B. Richerzhagen, J. Rückert, and D. Hausheer, *TRANSIT: Supporting transitions in peer-to-peer live video streaming*, in Proceedings of the IFIP Networking Conference, 2014, pp. 1–9.