Waypoint-based Topology Inference

Yilei Lin*, Ting He*, Shiqiang Wang†, Kevin Chan‡
*Pennsylvania State University, University Park, PA 16802, USA. Email: {yjl5282,tzh58}@psu.edu
†IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA. Email: wangshiq@us.ibm.com
‡US Army Research Laboratory, Adelphi, MD 20783, USA. Email: kevin.s.chan.civ@mail.mil

***Abstract*—Traditional network topology inference aims at reconstructing the routing trees rooted at each probing source from end-to-end measurements. However, due to emerging technologies such as network function virtualization, software defined networking, and segment routing, many modern networks are capable of supporting generalized forwarding that can create complex routing topologies different from routing trees. In this work, we take a first step towards closing this gap by proposing methods to infer the routing topology (referred to as 1-1-N topology) from a single source to multiple destinations, where routes may be required to traverse a given waypoint. We first thoroughly study the special case of 1-1-2 topologies, showing that even this seemingly simple case is highly nontrivial with 36 possibilities. We then demonstrate how the solution to the special case can be used as building blocks to infer 1-1-N topologies. The inferred topology is proved to be equivalent to the ground truth up to splitting/combining edges in the same category.**

***Index Terms*—Topology inference, waypoint-based routing, graphlet, 1-1-N topology.**

I. Introduction

Topology information is a critical part of network state that supports a variety of network management tasks such as service placement, routing, overlay construction, and load balancing. In single-domain networks where all the devices are owned and managed by the same provider, topology information can be directly collected, e.g., via the help of local monitoring agents running Simple Network Management Protocol (SNMP). In contrast, obtaining accurate topology information for public/shared networks such as the Internet is much more challenging due to the lack of local support.

Network tomography provides a promising approach to address this challenge by inferring the internal topology of a target network from end-to-end measurements collected from a subset of nodes. Since introduced in the 1990s to infer multicast trees from losses observed at multicast receivers [1], [2], the technique has been extended to exploit a variety of multicast and unicast measurements [3]–[6], and to infer more complicated topologies beyond multicast trees by jointly considering measurements from multiple sources [7]–[11]. However, all these works relied on a critical assumption that the monitored network employs *destination-based forwarding* along a set of predetermined (although unknown) routing trees.

This research was partly sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

While mostly satisfied in traditional IP networks, the assumption of tree-based routing is no longer valid in new networking paradigms such as Software Defined Networking (SDN) [12] and Network Function Virtualization (NFV) [13], where packets can be steered along non-tree routing topologies. In our previous work [14], we verified that for such networks, existing tree-based topology inference algorithms cannot even guarantee a feasible solution that is consistent with all the end-to-end measurements. Based on a two-step approach of first inferring the so-called *category weights* and then embedding the category weights into a constructed topology, we were able to guarantee that the constructed topology is always consistent with all the measurements. It is, however, not necessarily the same as the ground-truth topology, due to the existence of many equivalent topologies that could have generated the measurements.

In this work, we aim at improving the inference accuracy by developing new topology inference algorithms. To reduce the ambiguity caused by the existence of equivalent topologies, we impose the constraint that each path must traverse a predetermined sequence of *waypoints*, and the routing between waypoints must follow the default routing protocol (e.g., shortest path routing). Waypoint-based routing models a variety of practical scenarios that cannot be modeled by simple destination-based forwarding. For example, in networks employing NFV, a flow may need to traverse a chain of virtualized network functions (VNFs) before going to the destination. Moreover, in IP networks employing segment routing [15], an ingress node may prepend a list of segments to the header of each packet to specify the list of network elements the packet must traverse. In these examples, the VNFs and the segments serve as waypoints, which are connected by the default routing paths (e.g., IGP shortest paths).

To discover the routing topologies of such networks, we address, for the first time, topology inference under *waypoint-based routing*. As a first step towards solving this problem, we consider a simple setting where there is only one probing source, one waypoint, and N ($N \geq 2$) destinations, referred to as the case of *1-1-N topology*. We show that the introduction of a waypoint significantly complicates the problem, and even the simplest case of 1-1-2 topology (called *graphlet*) has 36 possibilities. We then demonstrate how to use the graphlets as building blocks to infer 1-1-N topologies for an arbitrary N.

A. Related Work

Topology inference using end-to-end measurements was initially studied for multicast probing [1], [2], where correlation among losses observed at multicast receivers is used to infer the multicast tree. Over the years, the technique was extended to exploit a variety of multicast measurements, including losses [3], delays [4], and a combination of losses and delays [5]. Due to the limited support of multicast, unicast-based solutions were developed, using stripes of back-to-back unicast packets to emulate a multicast [6].

Recent works focused on handling underlying topologies that are not trees [7]–[11], by utilizing measurements from multiple sources. Solutions in [7]–[9] constructed directed acyclic graphs (DAGs) by merging 2-by-2 topologies (i.e., *quartets*) depicting the connections between two sources and two destinations. Given 1-by-2 and 2-by-1 topologies, [10] presented a necessary and sufficient condition for the underlying topology to be identifiable and an algorithm to do so. With the additional requirement that internal nodes support network coding, DAGs can be inferred at improved accuracy and reduced complexity [11]. However, all the above solutions assumed that there is a single route for every source-destination pair, and the routes from each source or to each destination form a tree.

To our knowledge, the only existing work considering non-tree-based routing is our previous work [14]. It proposed a two-step approach to construct a weighted topology that is consistent with all the end-to-end measurements taken from an arbitrary underlying topology with arbitrary routing. The constructed topology, however, is not guaranteed to resemble the ground-truth topology. In this work, we aim at improving the inference accuracy by considering waypoint-based routing.

B. Summary of Contributions

Our contributions are four-fold:

1) We are the first to tackle the problem of topology inference from end-to-end measurements under waypoint-based routing.
2) We discover all the possibilities of 1-1-2 topologies (called graphlets), and prove that these topologies are uniquely identifiable based on multicast probes.
3) We demonstrate that graphlets can be used as building blocks to identify 1-1-N topologies for a general $N \geq 2$.
4) We validate the accuracy of our topology inference algorithm based on real ISP topologies.

II. Problem Formulation

A. Network Model

We model the network (routing) topology as an edge-weighted directed graph $\mathcal{G} = (V, E)$. Each vertex $v \in V$ represents a source, a destination, a waypoint, or a branching/joining point between at least two measurement paths. Consider the case where there is one probing source s, one waypoint f, and N ($N \geq 2$) destinations $\{t_1, \ldots, t_N\}$. We refer to such a topology as a 1-1-N topology. Each edge $e \in E$ represents a connection between two adjacent vertices, which may map to a sequence of links in the physical topology. Given an edge e, let u_e denote its weight, which can represent various performance metrics as explained in [14].

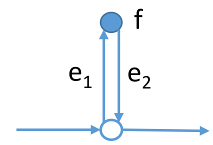


Fig. 1. Subgraph connecting waypoint f to the rest of the network ($u_{e_2} \equiv 0$).

We assume that the edge weight has the properties that (i) it is non-negative, and (ii) the sum weight over edges traversed by a multicast probe can be estimated consistently from end-to-end measurements. As a concrete example, consider the definition that $u_e := -\log \alpha_e$, where α_e is the probability for edge e to be in a "good" state (e.g., no loss, no queueing, no congestion). This weight is by definition non-negative. Moreover, if C denotes the set of paths probed in a given multicast, then the sum weight ϕ_C of edges in the multicast tree, referred to as the *cast weight*, equals

$$\phi_C := \sum_{e \in \bigcup_{p \in C} p} u_e = -\log\left(\prod_{e \in \bigcup_{p \in C} p} \alpha_e\right) = -\log(\Pr\{X_C = 1\}), \quad (1)$$

where $X_C \in \{0, 1\}$ is the indicator that all the destinations of paths in C receive a multicast probe with "good" performance, i.e., without loss if α_e denotes the no-loss probability, without queueing delay if α_e denotes the no-queueing probability, and without congestion if α_e denotes the no-congestion probability. Therefore, as the number of multicast probes sent on C increases, the estimated cast weight $\widehat{\phi}_C := -\log(\widehat{\Pr}\{X_C = 1\})$ ($\widehat{\Pr}\{X_C = 1\}$: empirical probability of $X_C = 1$) will converge to the true value. Similarly, for any subset of paths $A \subseteq C$, we can also measure the indicator X_A that all the destinations of paths in A receive a multicast probe with good performance to obtain a consistent estimate of $\phi_A := \sum_{e \in \bigcup_{p \in A} p} u_e$.

We assume that the waypoint f connects to the rest of the network via the subgraph in Fig. 1, i.e., a two-hop cycle. This is used to model cases where probes can incur nontrivial performance cost (e.g., loss, queueing) at the waypoint, e.g., when the waypoint represents a VNF. We model the performance cost due to processing at f by the weight of the incoming edge e_1, leaving the weight of e_2 to zero. In this way, all possible performance costs are modeled by edge weights. This structure also allows a probe to skip the waypoint in any topology.

B. Observation Model

Source s wants to learn the network topology $\mathcal{G}$. Other than the destinations and the waypoint, it cannot observe the internal vertices (i.e., branching/joining points), the edges, or the edge weights. Instead, it can probe the destinations, possibly via the waypoint. We refer to the source, the destinations, and the waypoint as *observable vertices*. We assume that the source can specify the destination for each measurement path and whether or not the path traverses the waypoint. However, routing between consecutively traversed observable vertices is uncontrollable and governed by the underlying routing protocol, which is assumed to follow deterministic destination-based forwarding (e.g., shortest path routing). We assume that the underlying routing protocol is consistent (i.e.,

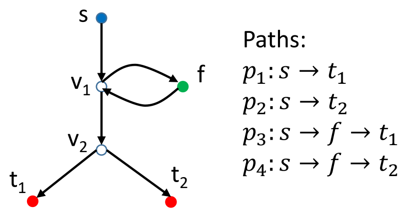


Fig. 2. Example of a possible 1-1-2 topology.

always taking the same path between a pair of vertices). We further assume that all the end-to-end paths are cycle-free except for the two-hop cycle containing the waypoint[1]. We denote the path traversing observable vertices $v_1, \ldots, v_n$ by $v_1 \to \ldots \to v_n$, e.g., the path from source s to destination t_j via waypoint f is $s \to f \to t_j$.

We further assume that the source can perform multicast probing, or approximated multicast probing via back-to-back unicast probes. It is known [6]that back-to-back unicast probes can emulate multicast probes in terms of end-to-end measurements. As explained in (1), the performances of these (approximated) multicast probes can be used to obtain consistent estimates of the cast weights, each being the sum of edge weights over any subset of the probed paths.

C. Topology Inference Problem

Our goal is to develop a probing-based topology inference algorithm to infer the 1-1-N topology.

1) Existing Results: We will leverage our previous work [16] to transform our observations from cast weights to quantities at a finer granularity, defined as follows.

Definition 1. Given a set C of paths probed in a multicast, we define the following:

1) a *category* $\Gamma_{\frac{A}{C}}$ for $A \subseteq C$ and $A \neq \emptyset$ is the set of edges traversed by and only by paths in A when the set of probed paths is C, i.e., $\Gamma_{\frac{A}{C}} := \{e \in E : e \in p$ for all $p \in A$, $e \notin p$ for all $p \in C \setminus A\}$;
2) the *category weight* for category $\Gamma_{\frac{A}{C}}$, denoted by $w_{\frac{A}{C}}$, is the sum weight for the edges in $\Gamma_{\frac{A}{C}}$.

Let $\mathcal{C} := 2^C \setminus \{\emptyset\}$ denote all the nonempty subsets of C.

The subscript "$\frac{A}{C}$" indicates that which edges belong to a category depends not only on A, but also on C. To simplify the notation, when a measurement path $s \to f \to t_j$ (or $s \to t_j$) appears in the subscript, we will denote it by (f, t_j) (or (t_j)), omitting the source s as it is implicit. However, we still need to specify the entire sequence of traversed observable vertices in other places to denote a subpath (e.g., $f \to t_j$) that does not start from s.

Theorem II.1 (Theorem III.1 in [16])**.** Given the cast weights $(\phi_A)_{A \in \mathcal{C}}$ obtained from multicast probes on C, all the category weights $(w_{\frac{A}{C}})_{A \in \mathcal{C}}$ can be uniquely determined.

Example: Consider the example in Fig. 2 with four possible measurement paths. Using multicast probing on all the four paths, we can consistently estimate the cast weight for any subset of these paths, e.g., the cast weight $\phi_{\{(t_1),(t_2)\}}$ gives the sum weight of edges (s, v_1), (v_1, v_2), (v_2, t_1), and (v_2, t_2). Edges (s, v_1) and (v_1, v_2) are in category $\Gamma_{\frac{\{(t_1),(t_2),(f,t_1),(f,t_2)\}}{\{(t_1),(t_2),(f,t_1),(f,t_2)\}}}$ as they are traversed by all the four paths, edges (v_1, f) and (f, v_1) are in category $\Gamma_{\frac{\{(f,t_1),(f,t_2)\}}{\{(t_1),(t_2),(f,t_1),(f,t_2)\}}}$, edge (v_2, t_1) is in category $\Gamma_{\frac{\{(t_1),(f,t_1)\}}{\{(t_1),(t_2),(f,t_1),(f,t_2)\}}}$, and edge (v_2, t_2) is in category $\Gamma_{\frac{\{(t_2),(f,t_2)\}}{\{(t_1),(t_2),(f,t_1),(f,t_2)\}}}$. Theorem II.1 says that we can uniquely determine all the category weights from the cast weights.

We will leverage this result by assuming the input of the topology inference algorithms to be the category weights for each category defined by a multicast. By Definition 1, edges in the same category are interchangeable in that they have the same influence on the measurements. Thus, we consider the ground truth $\mathcal{G}$ *identified* if the inferred topology is equivalent to $\mathcal{G}$ after splitting/combining edges in the same category (excluding edges to/from a waypoint as in Fig. 1).

III. Inference of 1-1-N Topology

We will first examine the special case of 1-1-2 topologies, and then demonstrate how these topologies can be used as building blocks to infer 1-1-N topologies. To ease presentation, we introduce a few notations in Table I.

TABLE I
NOTATIONS

T_N	$s \to t_1 \bigcup \ldots \bigcup s \to t_N$
T'_N	$s \to f \to t_1 \bigcup \ldots \bigcup s \to f \to t_N$
$B_{t_i t_j}$	the branching point[2]between $s \to t_i$ and $s \to t_j$
B_{fT}	the branching point between $s \to f$ and T_N
$B_{f t_j t_k}$	the branching point between $f \to t_j$ and $f \to t_k$
$J_{f t_j}$	the joining point between $f \to t_j$ and $s \to t_j$

A. 1-1-2 Topology

We first consider the special case with one source s, one waypoint f, and two destinations t_1, t_2, and we call this topology a *graphlet*. To infer the graphlet, we first infer a binary tree T_2 formed by the paths from the source directly to each of the two destinations without traversing any waypoint by an existing algorithm called *Rooted Neighbor-Joining (RNJ)* [6]. We then infer how f is connected to this tree by sending multicast probes on paths $s \to t_1$, $s \to t_2$, $s \to f \to t_1$, and $s \to f \to t_2$, indexed as $p_1, \ldots, p_4$, and measuring the category weights.

Theorem III.1. By sending multicast probes on paths $s \to t_1$, $s \to t_2$, $s \to f \to t_1$, $s \to f \to t_2$, the 1-1-2 topology (i.e., graphlet) can be uniquely identified if all the edges (except for the outgoing edge of f) have non-zero weights.

Proof. Based on all possible locations of the branching/joining points, we show in Lemma III.2 in [17] that there are 36 possible graphlets. Fig. 3 illustrates 20 of them, and there are 16 more graphlets that are symmetric counterparts

[1]While it is possible to construct waypoint-based routing paths that are not cycle-free, we note that such paths are undesirable for performance reasons and generally avoidable via proper waypoint placement.

[2]For a path p and a graph Q (which may be a path), their branching point is the last vertex they share, and their joining point is the first vertex they share.

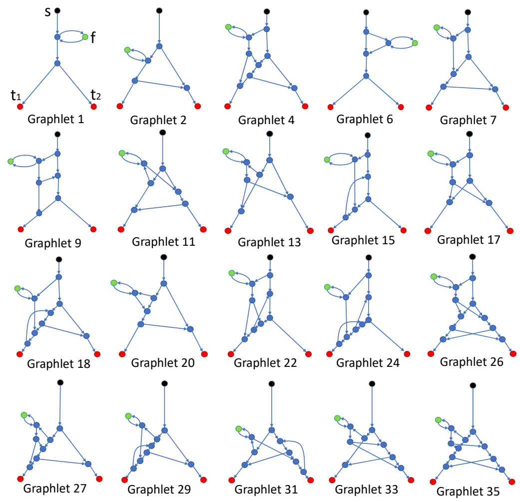


Fig. 3. Possible types of graphlets (excluding symmetric counterparts).

graphlet	{1}	{2}	{3}	{4}	{1,2}	{1,3}	{1,4}	{2,3}	{2,4}	{3,4}	{1,2,3}	{1,2,4}	{1,3,4}	{2,3,4}	{1,2,3,4}
1	0	0	0	0	0	1	0	0	1	2	0	0	0	0	2
2	0	1	0	1	0	1	0	0	1	2	0	0	2	0	1
3	1	0	1	0	0	1	0	0	1	2	0	0	0	2	1
4	2	1	1	2	1	1	1	0	1	4	0	0	0	0	1
5	1	2	2	1	1	1	0	1	1	4	0	0	0	0	1
6	0	0	0	0	1	1	0	0	1	4	0	0	0	0	2
7	1	1	0	1	1	1	0	0	1	4	0	0	1	0	1
8	1	1	1	0	1	1	0	0	1	4	0	0	0	1	1
9	1	0	1	1	1	1	0	0	1	4	0	1	0	0	1
10	0	1	1	1	1	1	0	0	1	4	1	0	0	0	1
11	1	1	1	0	0	1	0	0	1	4	0	0	1	1	1
12	1	1	0	1	0	1	0	0	1	4	0	0	1	1	1
13	1	1	1	1	0	1	0	0	1	4	0	0	1	0	1
14	1	1	1	1	0	1	0	0	1	4	0	0	0	1	1
15	1	0	0	1	1	1	0	0	1	4	0	1	1	0	1
16	0	1	1	0	1	1	0	0	1	4	1	0	0	1	1
17	1	1	1	1	1	1	0	0	1	4	0	0	0	0	1
18	2	1	0	2	1	1	1	0	1	4	0	0	1	0	1
19	1	2	2	0	1	1	0	1	1	4	0	0	0	1	1
20	1	1	0	1	0	1	0	0	1	4	0	0	2	0	1
21	1	1	1	0	0	1	0	0	1	4	0	0	0	2	1
22	2	0	1	2	1	1	1	0	1	4	0	1	0	0	1
23	0	2	2	1	1	1	0	1	1	4	1	0	0	0	1
24	2	0	0	2	1	1	1	0	1	4	0	1	1	0	1
25	0	2	2	0	1	1	0	1	1	4	1	0	0	1	1
26	2	2	2	2	1	1	1	1	1	4	0	0	0	0	1
27	2	1	1	2	0	1	1	0	1	4	0	0	1	0	1
28	1	2	2	1	0	1	0	1	1	4	0	0	0	1	1
29	2	1	0	2	0	1	1	0	1	4	0	0	2	0	1
30	1	2	2	0	0	1	0	1	1	4	0	0	0	2	1
31	1	2	2	0	0	1	0	1	1	4	0	0	1	1	1
32	2	1	0	2	0	1	1	0	1	4	0	0	1	1	1
33	1	2	2	1	0	1	0	1	1	4	0	0	1	0	1
34	2	1	1	2	0	1	1	0	1	4	0	0	0	1	1
35	2	2	2	2	0	1	1	1	1	4	0	0	1	0	1
36	2	2	2	2	0	1	1	1	1	4	0	0	0	1	1

Fig. 4. Number of edges in each category for each graphlet.

of some of the illustrated graphlets. Specifically, for graphlet 2, there exists a symmetric graphlet indexed as graphlet 3 that exchanges the locations of[3] t_1 and t_2. Similarly, for each of graphlets 4, 7, 9, 11, 13, 15, 18, 20, 22, 24, 27, 29, 31, 33, 35, there exits a symmetric graphlet, indexed as graphlet 5, 8, 10, 12, 14, 16, 19, 21, 23, 25, 28, 30, 32, 34, 36, respectively. Note that graphlets 1, 6, 17, and 26 are already symmetric with respect to (wrt) t_1 and t_2. By Theorem II.1, we can accurately estimate the category weights from sufficiently many multicast probes. Under the assumption that all the edges (except the outgoing edge of f) have non-zero weights, we can detect which categories are nonempty (i.e., there exists at least one edge in the ground truth graphlet in that category), as a category has a non-zero weight if and only if it is nonempty.

Fig. 4 lists the number of edges in each category in each graphlet. Here, as the set of probed paths is always $\{p_1, p_2, p_3, p_4\}$, we omit C and only specify A for each category in terms of path indices, e.g., column $\{1\}$ is the number of edges traversed by p_1 but no other path. We find that these 36 different types of graphlets correspond to 36 different combinations of nonempty categories. Therefore, we can uniquely determine the type of the ground truth graphlet from the inferred set of nonempty categories. □

We call edges in categories $\Gamma_{\frac{\{(t_1),(f,t_2)\}}{\{(t_1),(t_2),(f,t_1),(f,t_2)\}}}$ and $\Gamma_{\frac{\{(t_2),(f,t_1)\}}{\{(t_1),(t_2),(f,t_1),(f,t_2)\}}}$ (i.e., columns $\{1,4\}$ and $\{2,3\}$ in Fig. 4) *floating edges*, as the weights of the edges before and after them cannot be determined individually (see Step 3 in the proof of Theorem III.2 in [17] for details).

B. 1-1-N Topology

Now we will consider the topology with one source s, one waypoint f, and N ($N \geq 2$) destinations $t_1, \cdots, t_N$.

Theorem III.2. If all the edges (except for the outgoing edge of f) have non-zero weights, then the 1-1-N topology can be uniquely identified by sending multicast probes on up to four paths at a time, up to permutations of floating edges that are mapped onto the same edge in T_N.

Sketch of proof. Due to space limitation, we only provide a sketch of proof here and refer to [17] for detail.

By the assumption in Section II-B, the routing topology T_N from s to $t_1, \ldots, t_N$ without any waypoint is a tree, and thus can be inferred by the existing algorithm *Rooted Neighbor-Joining (RNJ)* [6]. Without loss of generality, we assume T_N to be a binary tree. We use M_l to denote a branching point in T_N. Starting from the root s, let the first branching point be M_1. Let the left child of M_l be M_{l1}, and the right child of M_l be M_{l2}.

Similarly, routes from f to $t_1, t_2, \cdots, t_N$ form a tree, denoted by T_f. Without loss of generality, we assume T_f to be a binary tree. We use M'_l to denote a branching point in T_f. Starting from the root f, let the first branching point be M'_1. Let the left child of M'_l be M'_{l1}, and the right child of M'_l be M'_{l2}. Paths $s \to f \to t_1, \ldots, s \to f \to t_N$ share the same subpath $s \to f$. Thus, we can still use RNJ [6] to infer T'_N (defined in Table I)[4], which is simply $(s \to f) \bigcup T_f$.

To infer the 1-1-N topology, it suffices to combine T'_N with T_N. We are going to consider the following steps: 1) determine where B_{fT} is located in T_N; 2) determine where each M'_l is located wrt T_N (possibly not in T_N); 3) determine how each M'_l that is adjacent to a destination in T_f is connected to the destination in the combined graph; 4) determine how neighboring branching points M'_l and M'_k are connected to each other in the combined graph. A basic building block is to infer the graphlet formed by paths $\{s \to t_i, s \to t_j, s \to f \to t_i, s \to f \to t_j\}$ from multicast

[3]Graphlet 3 is not equivalent to graphlet 2 because the destinations are observable vertices.

[4]Precisely, RNJ will give a tree equivalent to T'_N after contracting $s \to M'_1$ into a single edge.

probes on these paths as shown in the proof of Theorem III.1. We will denote the resulting graphlet by $\mathcal{G}_{ij}$ and its index (among the 36 possible graphlets) by R_{ij}.

Step 1). By definition, B_{fT} must lie on T_N. We start from M_1, choosing one (arbitrary) destination t_i from the left branch of M_1 and one destination t_j from the right branch of M_1. Then we send multicast probes to infer the graphlet index R_{ij} for this pair of destinations. If $R_{ij} \in \{1$, 4–10, 15–19, 22–26$\}$, then B_{fT} is above M_1 (i.e., between s and M_1). If $R_{ij} \in \{2, 11, 13, 20, 27, 29, 31, 33, 35\}$, then B_{fT} is on the left branch of M_1. If $R_{ij} \in \{3, 12, 14, 21, 28, 30, 32, 34, 36\}$, then B_{fT} is on the right branch of M_1. Because of the symmetry, we will only consider the case when B_{fT} is on the left branch. In this case, we find the left child of M_1, denoted by M_{11}, and choose one destination $t_{i'}$ from the left branch of M_{11} and one destination $t_{j'}$ from the right branch of M_{11}. Then we send multicast probes on paths $\{s \to t_{i'}, s \to t_{j'}, s \to f \to t_{i'}, s \to f \to t_{j'}\}$, and apply the above analysis to infer if B_{fT} is above M_{11} (i.e., between M_{11} and M_1), on the left branch of M_{11}, or on the right branch of M_{11}. Applying this method iteratively will uniquely locate B_{fT} onto an edge of T_N.

Step 2). To discover the position of each M'_l wrt T_N, we perform bottom-up probing, each time randomly choosing a pair of candidate vertices (t'_i, t'_j) that are siblings on T_f for probing. The candidate vertices include the destinations and the branching points in T_f that have been located wrt T_N (initially, the candidate vertices only include the destinations). Here, we treat a located branching point as a representative of the destinations under it. The selected probing destinations will be t_i and t_j, where $t_i = t'_i$ if t'_i is a destination, or an arbitrary destination under t'_i if t'_i is a branching point, and similarly for t_j. Then we send multicast probes on path $\{s \to t_i, s \to t_j, s \to f \to t_i, s \to f \to t_j\}$ to discover the corresponding graphlet $\mathcal{G}_{ij}$, with index R_{ij}. The above selection process ensures that the branching point $B_{ft_it_j}$ in $\mathcal{G}_{ij}$ is the parent of t'_i and t'_j in T_f.

We observe that R_{ij} helps to locate $B_{ft_it_j}$ wrt T_N. Specifically, if $R_{ij} \in \{1, 6\}$, we know that $B_{ft_it_j}$ coincides with $B_{t_it_j}$ (a branching point in T_N). If $R_{ij} \in \{2, 7, 12, 15, 18, 20, 24, 29, 32\}$, then $B_{ft_it_j}$ is on path $B_{t_it_j} \to t_i$, at a distance $w_{\frac{\{(t_i),(f,t_i)\}}{\{(t_i),(t_j),(f,t_i),(f,t_j)\}}}$ away from t_i, since edge $(B_{ft_it_j}, t_i)$ is the only edge in category $\Gamma_{\frac{\{(t_i),(f,t_i)\}}{\{(t_i),(t_j),(f,t_i),(f,t_j)\}}}$ in these graphlets. A symmetric case is when $R_{ij} \in \{3, 8, 11, 16, 19, 21, 25, 30, 31\}$, which uniquely localizes $B_{ft_it_j}$ at a point between $B_{t_it_j}$ and t_j. If $R_{ij} \in \{4$, 5, 9, 10, 13, 14, 17, 22, 23, 26–28, 33–36$\}$, then $B_{ft_it_j}$ may not be on[5] T_N.

Step 3). This step identifies the path $B_{ft_it_j} \to t_j$ for each pair of branching point $B_{ft_it_j}$ and destination t_j that are adjacent in T_f. Although path $B_{ft_it_j} \to t_j$ may join and branch from T_N multiple times, we haven shown in [17] that the positions of these joining and branching points can be almost uniquely determined.

Step 4). This step determines the paths between neighboring branching points of T_f. In [17], we provide a method to determine the path between a branching point $B_{ft_it_j}$ whose paths to destinations t_i and t_j are already determined, and a branching point $B_{ft_it_k}$ that is the parent of $B_{ft_it_j}$. Applying this method from the bottom of T_f up will determine the paths between all the neighboring branching points of T_f. □

[5]Precisely, $\mathcal{G}_{ij}$ implies that $B_{ft_it_j}$ is not on the tree $s \to t_i \bigcup s \to t_j$. It is possible that $B_{ft_it_j}$ is on $s \to t_k$ for some $t_k \neq t_i$ or t_j, or $B_{ft_it_j}$ is not on T_N at all. We will resolve this uncertainty in subsequent steps.

Algorithm 1: 1-1-N topology inference

input : Source s, waypoint f, destinations $t_1, \ldots, t_N$
output: 1-1-N topology $\mathcal{G}$

infer T_N and T'_N by algorithm RNJ in [6];
initialize $\mathcal{G}$ to T_N;
`Step 1):`
locate B_{fT} in $\mathcal{G}$ by Algorithm 2 in [17];
connect B_{fT} to f by the structure in Fig. 1;
`Step 2):`
locate branching points of T'_N wrt $\mathcal{G}$ by Algorithm 3 in [17];
`Step 3):`
construct paths in $\mathcal{G}$ to connect the vertices corresponding to bottom-level branching points in T'_N to the underlying destinations by Algorithm 4 in [17];
`Step 4):`
construct paths in $\mathcal{G}$ to connect the vertices corresponding to adjacent branching points in T'_N by Algorithm 5 in [17];
return $\mathcal{G}$;

---

1) Topology inference algorithm: The proof of Theorem III.2 already gives an algorithm to infer the 1-1-N topology, formalized in Algorithm 1. Theorem III.2 guarantees that the algorithm is accurate (up to permutations of floating edges), if all the edges that represent real paths or processing units have non-zero weights. As real paths or processing units inevitably incur some performance cost to probes traversing them, this condition should be satisfied with high probability with sufficiently many probes.

Remark: Algorithm 1 only performs multicast on up to four paths at a time (i.e., 4-cast). Compared to arbitrary multicast, k-cast for a small constant k has the advantage that it can be accurately emulated by back-to-back unicast probes [6], making it more applicable in practice.

IV. Performance Evaluation

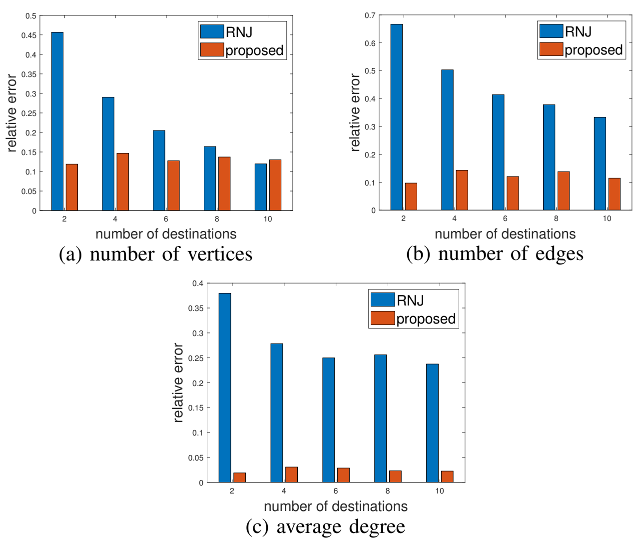
Benchmark: As we are the first to study topology inference under waypoint-based routing, we use a state-of-the-art topology inference algorithm designed for destination-based forwarding as the benchmark. The algorithm is called *Rooted Neighbor-Joining (RNJ)* [6], which infers a tree topology using bi-cast probing, and is guaranteed to be accurate when the ground truth topology is a canonical tree.

Simulation setting: Our simulation is conducted on Internet Service Provider (ISP) topologies from the Rocketfuel project [18], which are router-level topologies collected from diverse ISPs. We leave evaluations on newer datasets to future work. In our experiment, we choose topology AS6461, which represents the ISP Abovenet in US with 182 vertices and 294 edges. The weight of each edge is uniformly distributed in $[0.005, 0.05]$, which means that the probability for each edge to be in a “good” state (e.g., lossless) ranges from 95.12% to 99.50%.

In our simulation, we randomly choose a source and N destinations from nodes with degree ≤ 2, and a waypoint from nodes with degree ≥ 6. We use Dijkstra’s algorithm (based on hop count) to route between the selected vertices, which generates piecewise shortest paths. The ground truth is preprocessed to contract consecutive edges in the same category.All our results are averaged over 10 Monte Carlo runs.

Results: We measure the accuracy of reconstructing typical graph parameters, e.g., number of vertices, number of edges, and average degree, when varying the number of destinations. For each parameter ρ, we measure the relative error defined as $|\hat{\rho} - \rho|/\rho$, where $\hat{\rho}$ is the corresponding parameter in the inferred topology. The result, shown in Fig. 5, shows that the topology inferred by our algorithm is much closer to the ground truth than that by RNJ.

We have also examined the inferred topologies in detail. We find that graphlets 4, 5, 18, 19, and 22–36 are very rare in practice. Most errors of our algorithm are because the ground truth topology does not contain all the edges in a category. For example, in Fig. 6, the green edges in the inferred topology belong to the same category (also the same as the green edge in the ground truth), but one of them does not exist in the ground truth. Thus, combining these green edges (i.e., contracting the top edge and shifting its weight to the bottom edge) will make the inferred topology identical to the ground truth.



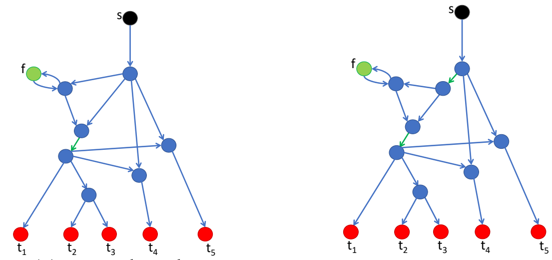
(a) number of vertices (b) number of edges

(c) average degree

Fig. 5. Accuracy of inferring 1-1-N topology.

V. Conclusion

We tackled, for the first time, the problem of topology inference under waypoint-based routing. We showed that using a novel notion called category weights that can be consistently estimated from end-to-end measurements, we could identify the 1-1-N topology depicting the connections between a source, a waypoint, and multiple destinations. Our algorithm was shown to be accurate (up to splitting/combining edges in the same category) both theoretically and empirically. Our result provides a stepping stone towards inferring more general topologies with multiple waypoints.



(a) ground truth (b) inferred topology

Fig. 6. Example: The inferred topology is equivalent to the ground truth after combining edges in the same category (in green).

References

[1] R. Caceres, N. G. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, “Loss-based inference of multicast network topology,” in *IEEE CDC*, 1999.

[2] S. Ratnasamy and S. McCanne, “Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements,” in *IEEE INFOCOM*, 1999.

[3] R. Bowden and D. Veitch, “Finding the right tree: Topology inference despite spatial dependences,” *IEEE Transactions on Information Theory*, vol. 64, no. 6, pp. 4594–4609, June 2018.

[4] S. Bhamidi, R. Rajagopal, and S. Roch, “Network delay inference from additive metrics,” *Journal of Random Structures & Algorithms*, vol. 37, no. 2, pp. 176–203, September 2010.

[5] N. G. Duffield, J. Horowitz, and F. L. Presti, “Adaptive multicast topology inference,” in *IEEE INFOCOM*, 2001.

[6] J. Ni, H. Xie, S. Tatikonda, and Y. R. Yang, “Efficient and dynamic routing topology inference from end-to-end measurements,” *IEEE/ACM Transactions on Networking*, vol. 18, no. 1, pp. 123–135, February 2010.

[7] M. Rabbat, M. Coates, and R. Nowak, “Multiple source Internet tomography,” *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2221–2234, December 2006.

[8] P. Sattari, M. Kurant, A. Anandkumar, A. Markopoulou, and M. G. Rabbat, “Active learning of multiple source multiple destination topologies,” *IEEE Transactions on Signal Processing*, vol. 62, no. 8, pp. 1926–1937, April 2014.

[9] A. Anandkumar, A. Hassidim, and J. Kelner, “Topology discovery of sparse random graphs with few participants,” in *ACM SIGMETRICS*, June 2011.

[10] G. Berkolaiko, N. Duffield, M. Ettehad, and K. Manousakis, “Graph Reconstruction from Path Correlation Data,” *Inverse Problems*, vol. 35, no. 1, 2018.

[11] P. Qin, B. Dai, B. Huang, G. Xu, and K. Wu, “A survey on network tomography with network coding,” *IEEE Communication Survey & Tutorials*, vol. 16, no. 4, pp. 1981–1995, 2014.

[12] “Software-defined networking: the new norm for networks,” Open Networking Foundation White Paper, April 2012.

[13] “Network Functions Virtualisation — Introductory White Paper,” White Paper, ETSI, 2012. [Online]. Available: https://portal.etsi.org/nfv/nfv_white_paper.pdf

[14] Y. Lin, T. He, S. Wang, K. Chan, and S. Pasteris, “Looking glass of NFV: Inferring the structure and state of NFV network from external observations,” in *IEEE INFOCOM*, April 2019.

[15] “Segment Routing Architecture,” RFC8402. [Online]. Available: https://www.rfc-editor.org/rfc/rfc8402.txt

[16] Y. Lin, T. He, S. Wang, K. Chan, and S. Pasteris, “Multicast-based weight inference in general network topologies,” in *IEEE ICC*, May 2019.

[17] Y. Lin, T. He, S. Wang, and K. Chan, “Waypoint-based topology inference,” Technical Report, October 2019. [Online]. Available: https://sites.psu.edu/nsrg/files/2020/01/WaypointTopologyInference.pdf

[18] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, “Measuring ISP topologies with Rocketfuel,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 1, pp. 2–16, February 2004.