Looking Glass of NFV: Inferring the Structure and State of NFV Network from External Observations

Yilei Lin, *Student Member, IEEE*, Ting He, *Senior Member, IEEE,* Shiqiang Wang, *Member, IEEE*, Kevin Chan, *Senior Member, IEEE,* and Stephen Pasteris

***Abstract*—The rapid development of network function virtualization (NFV) enables a communication network to provide in-network services using virtual network functions (VNFs) deployed on general IT hardware. While existing studies on NFV focused on how to provision VNFs from the provider's perspective, little is done about how to validate the provisioned resources from the user's perspective. In this work, we take a first step towards this problem by developing an inference framework designed to "look into" the NFV network. Our framework infers the structure and state of the overlay formed by VNF instances, ingress/egress points of measurement flows, and critical points on their paths (branching/joining points). Our solution only uses external observations such as the required service chains and the end-to-end performance measurements. Besides the novel application scenario, our work also fundamentally advances the state of the art on topology inference by considering (i) general topologies with general measurement paths, and (ii) information of service chains. Our evaluations show that the proposed solution significantly improves both the reconstruction accuracy and the inference accuracy over existing solutions, and service chain information is critical in revealing the structure of the underlying topology.**

***Index Terms*—Network function virtualization, network topology inference, network tomography**

I. Introduction

Modern communication networks have outgrown simple bit pipes. Increasingly, network providers use network appliances to provide in-network services, e.g., Network Address Translators (NATs), firewalls, Intrusion Detection Systems (IDSs), Intrusion Prevention Systems (IPSs), Deep Packet Inspectors (DPIs), web proxies, and WAN optimizers [2]. While traditionally deployed as physical middleboxes implemented by special-purpose hardware, next-generation network appliances are increasingly deployed as software middleboxes, referred to as *Virtual Network Functions (VNFs)*, running on general-purpose servers. This technology, known as Network Function Virtualization (NFV) [3], is empowering network providers to partner with cloud providers and software vendors to provide innovative value-adding services within the communication network [4].

Y. Lin (yjl5282@psu.edu) and T. He (tzh58@psu.edu) are with the Pennsylvania State University. S. Wang (wangshiq@us.ibm.com) is with IBM T. J. Watson Research Center. K. Chan (kevin.s.chan.civ@mail.mil) is with US Army Research Laboratory. S. Pasteris (s.pasteris@cs.ucl.ac.uk) is with University College London.

A preliminary version of this work was presented at INFOCOM'19 [1].

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.

On one hand, NFV opens up a whole new solution space for configuring the network. Encapsulated as virtual machine (VM) instances, VNFs can be scaled up/down, replicated, and/or migrated to suit the current demands. Moreover, multiple VNFs can be organized into a chain (a.k.a. *service chain*) to serve flows with multiple processing needs. Solutions have been developed to exploit the enlarged solution space from the provider's perspective, by optimizing the placement of VNFs [5]–[8], the routing among VNFs [9], or a combination of these actions [10]–[13].

On the other hand, the presence of (virtual or physical) network appliances significantly complicates network management. Due to the widespread deployment of network appliances, the network administrator needs to manage not only routers and switches, but also a variety of network appliances, leading to high operational expenses and administrative headaches [2]. The problem remains even with the virtualization of network appliances, as the network administrator still needs to manage VNFs based on software that is often developed by independent vendors [4]. Furthermore, as NFV becomes widely adopted by network providers, there will be needs for a client (network administrator) to validate the service received from its network provider, or for a network provider to validate the service received from its peers. It is therefore highly desirable to have a method that can "look into" the NFV network without directly measuring individual routers or VNF instances. Besides service validation, such a method can also be used to engineer the optimal use of the network, e.g., via service placement, flow scheduling, client-server association, and load balancing.

In this work, we take a first step towards addressing this problem by jointly inferring the internal structure and state of an NFV network using external observations. We consider two types of observations: (i) parameters of flow demands (e.g., ingress/egress points and service chains) and (ii) end-to-end performance measurements (e.g., delays and losses). While these observations do not directly specify the physical network topology, we argue that they can provide useful information about the VNF overlay, such as: the traversal of VNF instances by each flow, the sharing of links & VNF instances across flows, and the performance of these links & VNF instances. We note that while motivated by NFV networks, our models and solutions are applicable to any network employing generalized forwarding under the constraint of waypoint traversals.

We model the above information by a directed, vertex-labeled, and edge-weighted graph, referred to as the *VNF topology*, where the graph topology represents the interconnections between VNF instances, the vertex labels represent

the (logical) VNF placement, and the edge weights represent the VNF performances[1]. We refer to the problem of inferring the VNF topology as the *VNF topology inference problem*.

A. Related Work

NFV resource management: From an application perspective, our work is related to resource management in NFV. As an emerging technology, NFV resource management has only begun to be studied recently, where existing works have addressed VNF placement [5]–[8], admission control and path selection [9], and joint optimizations of multiple control knobs [10]–[13]. Specifically, [10], [11] jointly optimize VNF placement and routing, [12] jointly optimizes VNF placement, routing, and admission control under hard capacity constraints, and [13] jointly optimizes VNF placement, routing, and resource allocation under soft capacity constraints. However, all the above are from the provider's perspective. *To our knowledge, we are the first to investigate the validation of VNF provisioning from the user's perspective.*

Network topology inference: Technically, our work belongs to the family of works on topology inference using end-to-end measurements. In the context of communication networks, the problem was initially studied for multicast probing [14], [15], 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 [16]–[19], delays [19]–[21] and a combination of losses and delays [22]. Meanwhile, due to the limited support of multicast, unicast-based solutions [23]–[25] were developed, using stripes of back-to-back unicast packets [23], [24] or "sandwiches" of small and large packets [25]. Most of these algorithms are inspired by *phylogenetic tree algorithms*, which aim at constructing a tree-structured model to represent the measured distances between leaf nodes [26]. To improve the accuracy of topology inference, a hybrid approach was proposed in [27], where end-to-end measurements are augmented with a small number of direct measurements (obtained by traceroute).

A few works considered underlying topologies that are not trees [28]–[37], all based on measurements from multiple sources. However, solutions in [33], [35] still constructed tree topologies, except that the accuracy was analyzed with respect to an underlying topology that may not be a tree. Solutions in [28]–[31], [34] constructed directed acyclic graphs (DAGs) by merging 2-by-2 topologies (i.e., *quartets*) depicting the connections between two sources and two destinations, and a similar idea was used in [36] by merging 1-by-3 topologies. With the additional requirement that internal nodes support network coding, [32] constructed DAGs with reduced probing overhead. Assuming measurements of 1-by-2 and 2-by-1 topologies, [37] presented a necessary and sufficient condition for the underlying topology to be identifiable and an algorithm to do so. However, all the above solutions assumed that there is a single route for every source-destination pair, and the routes from/to each node form a tree. In the context of NFV, the requirements of VNF traversals can cause flows to deviate from the default routes (e.g., shortest paths), and hence the underlying topology may not be a tree even if all the probes are sent by a single source. *To our knowledge, our work is the first to investigate topology inference based on end-to-end measurements for an arbitrary topology under arbitrary routing.*

[1]More precisely, the weight of an edge $(s(e), t(e))$ represents the overall performance for data transfer from $s(e)$ to $t(e)$ and data processing at $t(e)$, the physical meaning of which will be explained in Section II-C.

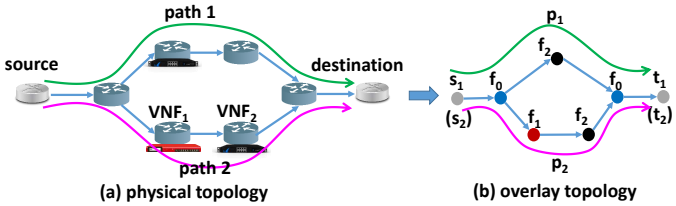


Fig. 1. Topologies of the physical substrate and the VNF overlay.

B. Summary of Contributions

The main contributions of this work are:

1) We are the first to consider external observation-based topology inference in NFV networks.

2) We show that the approach of tree approximation (for each probing source/destination) as used by existing topology inference algorithms is insufficient for NFV networks, and we propose a two-step solution framework, designed to give the smallest logical topology that is equivalent to the ground truth.

3) We extend our solution to incorporate information from service chains, by reformulating our problem as a novel combinatorial optimization problem (*string augmentation problem*) that can be cast as an integer linear program (ILP).

4) Through extensive data-driven simulations, we verify that our solution significantly outperforms existing solutions in both reconstructing the measurements and approximating the ground truth topology, where service chain information plays a critical role.

Roadmap. Section II formalizes our problem. Section III addresses a simplified version of our problem in a classical setting, and Section IV addresses the full version that incorporates service information. Section V evaluates the proposed solution against benchmarks. Section VI concludes the paper.

II. Problem Formulation

A. Network Model

As illustrated in Fig. 1, we model the VNF overlay by a directed, vertex-labeled, and edge-weighted graph $\mathcal{G} = (V, E, l, w)$, referred to as the *VNF topology*.

The vertex set V denotes the set of VNF instances and critical points on measurement paths (sources, destinations, and branching/joining points). The edge set E denotes the connections between these points, where for each edge $e \in E$, $s(e)$ denotes the starting point of this edge and $t(e)$ denotes the ending point. Map $l : V \to \mathcal{F}$ is a map from vertices to their labels that represent the VNF placement, where $l_v \in \mathcal{F}$ denotes the type of VNF placed at vertex $v \in V$, and $\mathcal{F} = \{f_1, f_2, \ldots\}$ denotes the set of all types of VNFs supported by the network. As measurement paths may start/end/branch/join at a vertex that does not run any VNF (e.g., a traditional packet switch), we introduce a dummy VNF $f_0 \notin \mathcal{F}$ to label such vertices. Lastly, $w : E \to \mathbb{R}$ represents the edge weights, where w_e for edge $e \in E$ models the overall performance

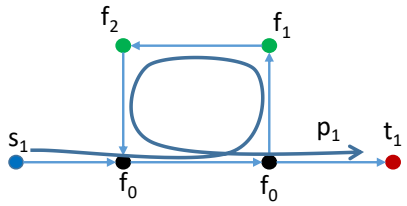


Fig. 2. Example of non-simple path: p_1 is the path for a flow from s_1 to t_1 traversing service chain (f_1, f_2) (i.e., going from s_1 to f_1, then to f_2, and then to t_1).

in transferring a packet from $s(e)$ to $t(e)$ *and* processing the packet at $t(e)$. In this work, we consider a family of performance metrics that can be modeled as additive edge weights as detailed in Section II-C. Note that $\mathcal{G}$ is not directly observable to the inference engine.

Note that $\mathcal{G}$ is by definition a logical topology that models the VNF overlay. It is known that the physical network topology cannot be inferred from external observations [34], but the logical topology still contains valuable information about the network structure and performance.

B. Flow Model

We measure the network by monitoring a set of flows $\mathcal{D} = \{d_i\}_{i=1}^n$, each demanding a source (or ingress point) s_i, a destination (or egress point) t_i, and a service chain $\mathbf{c}_i = (c_{i,j})_{j=1}^{n_i}$, where $c_{i,j} \in \mathcal{F}$ is the type of VNF required at step j of processing flow d_i. As the flow demands are provided by the users (or a proxy of the users), they are assumed to be observable to the inference engine. After a flow d_i is admitted by the network, it is mapped onto a path p_i that goes from s_i to t_i and traverses the service chain $\mathbf{c}_i$ in between. The internal portion of p_i (other than s_i and t_i) is not observable to the inference engine. Note that due to the traversal of a service chain, a flow may follow a *non-simple path*, which may traverse a vertex/edge multiple times as illustrated in Fig. 2.

C. Performance Model

We consider a family of edge weights with two properties: (i) the weights are nonnegative and additive, i.e., the path weight equals the sum weight of the traversed edges, and (ii) the weights can be reliably inferred (by an unbiased estimator) from end-to-end measurements for each path and the shared portion of each pair of paths. Let ρ_i denote the sum of edge weights on path p_i, referred to as the *path length*, and ρ_{ij} denote the sum of edge weights on the shared portion of paths p_i and p_j, referred to as the *shared path length*. It is known [23] that several important performance metrics satisfy these properties, briefly reviewed below. Our formulation can use any of these metrics.

We use a "probe" to refer to a packet sent on one of the measurement paths $\{p_i\}_{i=1}^n$ to obtain an end-to-end measurement. As in [23], [38], we assume that probes are sent in back-to-back pairs on a pair of paths at a time, so that probes in the same pair experience the same performances at shared edges. Nevertheless, the definitions below can be modified to account for imperfect correlation at shared edges [23]. We further assume that an edge performs independently for different probe pairs, and different edges perform independently. It is known that a stripe of back-to-back unicast probes can emulate a multicast probe [23]–[25]. In particular, bi-cast probes emulated by pairs of back-to-back unicast probes are known to be sufficient for accurately inferring the routing topology rooted at each source if the topology is a tree [23]. We thus assume this probing mechanism to examine its capability in inferring general topologies under arbitrary routing.

1) Loss-based Weight: If we measure the end-to-end losses, then the edge weight can be defined as $w_e := -\log \alpha_e$, where α_e is the success rate of edge e (i.e., the probability for a probe to successfully traverse edge e and get processed by the VNF at vertex $t(e)$). Let X_p be the success indicator for path p. Then we have

$$\rho_i = \sum_{e \in p_i} -\log \alpha_e = -\log \Pr\{X_{p_i} = 1\}, \tag{1}$$

$$\rho_{ij} = \sum_{e \in p_i \cap p_j} -\log \alpha_e = -\log \left(\frac{\Pr\{X_{p_i} = 1\} \Pr\{X_{p_j} = 1\}}{\Pr\{X_{p_i} = X_{p_j} = 1\}} \right). \tag{2}$$

Thus, we can calculate the path lengths and the shared path lengths by estimating the success probability of each path and the joint success probability for each pair of paths from the end-to-end losses. It is known that the unbiased estimators of these probabilities are simply their empirical values.

2) Utilization-based Weight: If we measure the end-to-end delays, then the edge weight can be defined as $w_e := -\log \beta_e$, where β_e is the no-queueing probability of edge e (i.e., the probability that a probe incurs no queueing delay in traversing edge e and getting processed at vertex $t(e)$). Let Y_p be the no-queueing indicator for path p. Then we have

$$\rho_i = \sum_{e \in p_i} -\log \beta_e = -\log \Pr\{Y_{p_i} = 1\}, \tag{3}$$

$$\rho_{ij} = \sum_{e \in p_i \cap p_j} -\log \beta_e = -\log \left(\frac{\Pr\{Y_{p_i} = 1\} \Pr\{Y_{p_j} = 1\}}{\Pr\{Y_{p_i} = Y_{p_j} = 1\}} \right). \tag{4}$$

Similar to loss-based weights, we can calculate the path lengths and the shared path lengths by estimating the no-queueing probabilities of each path and each pair of paths from end-to-end queueing indicators. In practice, this can be achieved by comparing each delay measurement with a threshold representing the "maximum end-to-end delay" on that path without queueing (estimated from delays measured when the network is lightly loaded), and counting the fraction of measurements below the threshold.

Remark: There is another "additive metric" defined in [23] that represents an edge weight by the variance of edge delay. We point out, however, that this definition is invalid in general when a path can traverse an edge multiple times, as the path weight no longer equals the sum weight of the traversed edges, e.g., an edge with delay variance w_l traversed twice by path p will contribute $4w_l$ to the delay variance of p.

3) Inferring Resource Provisioning from Weights: Under mild assumptions, we can use the edge weight to infer the amount of resources provisioned for the corresponding hop. As a concrete example, consider using an M/M/1/B queue to model each edge e (including data transfer on e and processing at $t(e)$). Let λ_e denote the arrival rate and μ_e denote the service rate, both measured by probes per unit time. Let $\sigma_e := \lambda_e/\mu_e$.

Let Q_e denote the queue length. It is known that the steady-state distribution of Q_e is $\pi_i = (1-\sigma_e)\sigma_e^i/(1-\sigma_e^{B+1})$

$(i = 0, \ldots, B)$. Therefore, the success (i.e., no queue overflow) probability α_e equals

$$\alpha_e = \Pr\{Q_e < B\} = \frac{1 - \sigma_e^B}{1 - \sigma_e^{B+1}}, \tag{5}$$

and the no-queueing probability β_e equals

$$\beta_e = \Pr\{Q_e = 0\} = \frac{1 - \sigma_e}{1 - \sigma_e^{B+1}}. \tag{6}$$

Given the inferred weight for edge e, we can easily calculate α_e in the case of loss-based weight and β_e in the case of utilization-based weight. Given the flow rates (observed) and the flow paths (inferred), we can calculate λ_e. If B is known, we can solve equations (5,6) for the service rate μ_e. If B is unknown, we can use the bounds ($\underline{B}$: lower bound on B)

$$\alpha_e \geq \frac{1 - \sigma_e^{\underline{B}}}{1 - \sigma_e^{\underline{B}+1}}, \tag{7}$$

$$\beta_e \geq 1 - \sigma_e \tag{8}$$

to compute lower bounds on σ_e and hence upper bounds on μ_e. Note that the lefthand sides of (7,8) are decreasing in σ_e. The inferred service rate μ_e (or its upper bound) thus characterizes the amount of resources provisioned for data transfer at e and data processing at $t(e)$.

D. VNF Topology Inference Problem

Given observations from a set of flows $\{d_i\}_{i\in[n]}$ ($[n] := \{1, \ldots, n\}$), including the sources, the destinations, the service chains, and the corresponding path lengths $\{\rho_i\}_{i\in[n]}$ and shared path lengths $\{\rho_{ij}\}_{i,j\in[n]}$, we want to infer the underlying VNF topology and the paths of these flows.

Topology Selection Criteria: The solution to the VNF topology inference problem will not be unique, e.g., dummy VNFs can be added without changing the service chains, and the sum weight of two edges traversed by the same set of paths can be split arbitrarily between them without affecting path lengths or shared path lengths. This is an inherent limitation of topology inference problems [34], [39]. To resolve the ambiguity, additional criteria are needed. Theoretically, the optimal solution should maximize the likelihood of the given measurements [25], [40]. In practice, however, simpler criteria are often used to avoid requiring statistical knowledge of the measurements (i.e., the likelihood function). In this work, we adopt a set of such nonparametric criteria.

Generally, given a set of feasible topologies, each consistent with all the observations, we want to select the topology that is:

1) a *minimum weight representation* that minimizes the total edge weight, or
2) a *minimum size representation* that minimizes the number of edges, or
3) a *minimum order representation* that minimizes the number of vertices.

Intuitively, (1) represents the "best-performing" topology in terms of the total weight, and (2–3) represent the "simplest" topology in terms of the number of edges or vertices. Any topology inference algorithm can only reconstruct the ground truth up to its minimum weight/size/order representation.

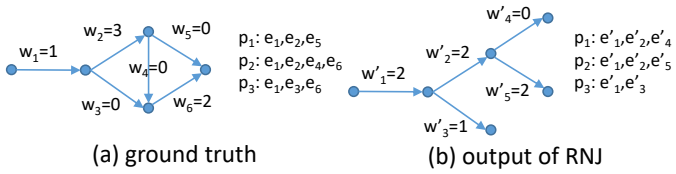


Fig. 3. Counterexample for tree approximation (w_i: the weight of edge e_i).

Remark: Objective (1) is consistent with the minimum spanning/Steiner tree model commonly used in phylogenetic inference [41], where the goal is to use a minimum weight tree to convey the relationships between species. Objective (2) is consistent with the penalized likelihood criterion in [25] (where the penalty is the number of edges) and the notion of "simplest topology" in [36]. Objective (3) is the same as the "minimal representation" criterion used in [34], where the goal is to reconstruct the measured distances between participating nodes using a minimum number of hidden nodes. We have renamed these criteria in the convention of graph theory.

III. Solutions Based on Path Length Information

We begin with a simplified version of the problem, where only path lengths and shared path lengths are used for inference as in [23], [37]. Accordingly, the goal is to infer a directed, edge-weighted graph $\mathcal{G} = (V, E, w)$, such that the flows can be mapped to paths in this graph that match the given path lengths and shared path lengths.

While this formulation ignores the knowledge of the sources/destinations, we can easily incorporate this knowledge by extending the inferred path of each flow to connect from/to its source/destination with zero-weight edges.

A. Deficiency of Tree Approximation

Although the simplified problem has been studied outside the context of NFV, existing solutions assumed that the underlying topology is either a tree or a union of trees (see Section I-A), neither valid in the context of NFV. Our first result is the following observation:

Claim III.1. It is not always possible to match the path/shared path lengths in a general ground truth topology by only constructing trees or unions of trees.

To illustrate this point, consider the ground truth topology in Fig. 3 (a). Ignoring measurement errors, we will observe the following: $\rho_1 = 4$, $\rho_2 = 6$, $\rho_3 = 3$, $\rho_{12} = 4$, $\rho_{13} = 1$, and $\rho_{23} = 3$. As all the paths start from a single source, all the existing solutions will attempt to use an edge-weighted tree rooted at the source to reconstruct these lengths. In particular, the *Rooted Neighbor-Joining (RNJ)* algorithm [23] guarantees correct reconstruction if the ground truth topology is a canonical tree and there is no measurement error. In this case, it returns a topology in Fig. 3 (b), which differs from the ground truth. Furthermore, the inferred topology does not match the measurements, as $\rho'_{13} = 2 \neq \rho_{13}$ and $\rho'_{23} = 2 \neq \rho_{23}$, i.e., it is not even a feasible solution.

We note that this is not just a limitation of RNJ: any tree topology will require at least two of the three shared path lengths to be equal, which is inconsistent with the measurements. Thus, this example illustrates *a fundamental*

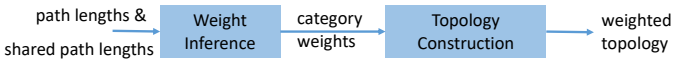


Fig. 4. The two-step solution framework.

limitation of tree approximation, indicating the need of a new topology inference algorithm that can construct arbitrary non-tree topologies.

B. Inference of General Topologies

We propose a solution framework for inferring a general topology based on path lengths and shared path lengths. Our framework consists of two steps as illustrated in Fig. 4: (1) weight inference, and (2) topology construction, where step (1) aims at inferring edge weights at the finest granularity (called *category weights*), and step (2) aims at constructing a graph based on the inferred weights to route the flows.

1) Weight Inference: We start by trying to infer edge weights based on the given path length information. Despite the unknown topology, we show that it is still possible to deduce weights at a finer granularity than paths/shared paths.

Problem Definition: We partition the edges into $2^n - 1$ *categories* (n: number of flows), where each category Γ_A for $A \subseteq [n]$ and $A \neq \emptyset$ (recall that $[n] := \{1, \dots, n\}$) contains the set of edges traversed by *exactly* the set of paths $\{p_i : i \in A\}$, i.e., $\Gamma_A := (\bigcap_{i\in A} p_i) \setminus (\bigcup_{i\notin A} p_i)$ (viewing each p_i as the set of edges on path p_i). Let w_A denote the *category weight* of Γ_A, i.e., the sum weight for all the edges in Γ_A. Let $\mathcal{A} := 2^{[n]} \setminus \emptyset$ denote all the category indices.

For example, for $n = 3$, we have a total of 7 categories, representing edges traversed by $\{p_1\}$, $\{p_2\}$, $\{p_3\}$, $\{p_1, p_2\}$, $\{p_1, p_3\}$, $\{p_2, p_3\}$, and $\{p_1, p_2, p_3\}$, respectively. In the ground truth topology in Fig. 3 (a), e_1 is in category $\Gamma_{\{1,2,3\}}$ as it is traversed by p_1, p_2, and p_3. Similarly, e_2 is in $\Gamma_{\{1,2\}}$, e_3 in $\Gamma_{\{3\}}$, e_4 in $\Gamma_{\{2\}}$, e_5 in $\Gamma_{\{1\}}$, e_6 in $\Gamma_{\{2,3\}}$, and $\Gamma_{\{1,3\}} = \emptyset$ (i.e., no edge is traversed by only p_1 and p_3).

Definition 1. The *weight inference problem* aims at determining the category weights $(w_A)_{A\in\mathcal{A}}$ from the given path lengths and shared path lengths.

We note that the path lengths and the shared path lengths only specify edge weights up to their sum within each category, as one can split each w_A arbitrarily among edges in category Γ_A without affecting any path length or shared path length. In this sense, the weight inference problem aims at inferring the edge weights at the finest granularity.

By definition, category-A edges are traversed by a path p_i if and only if $i \in A$. Similarly, category-A edges are shared by paths p_i and p_j if and only if $\{i, j\} \subseteq A$. Therefore, we can formulate the problem as solving the linear equations:

$$\sum_{A\in\mathcal{A}:i\in A} w_A = \rho_i, \qquad \forall i \in [n], \tag{9a}$$

$$\sum_{A\in\mathcal{A}:\{i,j\}\subseteq A} w_A = \rho_{ij}, \qquad \forall i, j \in [n], \tag{9b}$$

subject to (s.t.) the constraint that $w_A \geq 0$ ($\forall A \in \mathcal{A}$) due to the nonnegativity of edge weights (see Section II-C).

Challenges: There are several practical challenges in solving (9). First, there are exponentially many variables, suggesting that solving this linear system will incur *exponential complexity*. Moreover, there is only a quadratic number of equations, and thus we generally have an under-constrained linear system that *does not have a unique solution*. Furthermore, in practice we can only estimate the values of ρ_i's and ρ_{ij}'s from raw measurements, and the estimation errors can cause the linear system to be *infeasible*.

Results: For the first challenge, we first note that for each input, there is a solution where majority of the variables are zero.

Proposition III.2. For each topology, there exists a feasible solution to the weight inference problem that is $(n + \binom{n}{2})$-sparse, i.e., containing at most $n + \binom{n}{2}$ non-zero variables. Moreover, there exists a solution with the minimum total weight that is $(n + \binom{n}{2})$-sparse.

Proof. We note that the entire set of feasible solutions given by (9) and $w_A \geq 0$ ($\forall A \in \mathcal{A}$) is a bounded nonempty polytope in $\mathbb{R}^{2^n-1}$ space. Every vertex of this polytope, which is a feasible solution, is given by a subset of $2^n - 1$ constraints, where the inequality constraint $w_A \geq 0$ is satisfied with equality. As at least $2^n - 1 - n - \binom{n}{2}$ of these constraints are of the form $w_A = 0$, at most $n + \binom{n}{2}$ variables can be non-zero, i.e., feasible solutions corresponding to vertices of the polytope are $(n+\binom{n}{2})$-sparse. The second claim follows from the fact that if we further minimize $\sum_{A\in\mathcal{A}} w_A$ over the polytope, optimality can always be achieved at a vertex, which gives a minimum weight solution that is $(n + \binom{n}{2})$-sparse. □

Meanwhile, we have shown that no variable can be ignored (i.e., set to zero) agnostic to the input.

Proposition III.3. For each $A \in \mathcal{A}$, there exists an underlying topology for which w_A must be positive.

Proof. We prove the claim by contradiction. Suppose that there exists a weight inference algorithm π that always sets $w_A \equiv 0$ for all inputs. Consider a ground truth topology where only one edge in category Γ_A has a non-zero weight of 1; other edge weights are zero. Thus, $\rho_i = 1$ if $i \in A$, and $\rho_i = 0$ otherwise; $\rho_{ij} = 1$ if $\{i, j\} \subseteq A$, and $\rho_{ij} = 0$ otherwise. Let $\mathcal{A}'$ be the set of categories assigned non-zero weights by π. We argue that $\bigcup_{A'\in\mathcal{A}'} A'$ must equal A. Otherwise, we must have either (i) $i \in \bigcup_{A'\in\mathcal{A}'} A' \setminus A$, for which $\sum_{A':i\in A'} w_{A'} > 0$ but $\rho_i = 0$, or (ii) $i \in A \setminus (\bigcup_{A'\in\mathcal{A}'} A')$, for which $\sum_{A':i\in A'} w_{A'} = 0$ but $\rho_i = 1$. If $|A| = 1$, then $\mathcal{A}' = \{A\}$, i.e., π assigns a non-zero weight to category Γ_A, contradicting our assumption. If $|A| > 1$, we argue that for any $\{i, j\} \subseteq A$, $\nexists A' \in \mathcal{A}'$ that contains i but not j, because otherwise we must have $\rho_i > \rho_{ij}$. It implies that $\mathcal{A}' = \{A\}$, again contradicting our assumption. □

By Proposition III.2, once we know which subset of $n+\binom{n}{2}$ variables are non-zero, we can ignore the other variables and solve (9) in a time that is polynomial in n. However, Proposition III.3 implies that none of the $O(2^n)$ variables can be ignored (i.e., set to zero) for all the inputs. It remains open whether given an input, one can find, in polynomial time, a polynomial number of variables (i.e., category weights) such that there exists a feasible solution to (9) that assigns positive values only to these variables.

To address the second and the third challenges, we first relax the requirements from perfect reconstruction as in (9) to best-effort reconstruction, formulated as a constrained optimization:

$$\min \sum_{i\in[n]} |\sum_{A:i\in A} w_A - \rho_i| + \sum_{i,j\in[n]} |\sum_{A:\{i,j\}\subseteq A} w_A - \rho_{ij}| \quad (10a)$$

$$\text{s.t. } w_A \geq 0, \qquad \forall A \in \mathcal{A}. \quad (10b)$$

This is a convex optimization that can be solved by convex optimization solvers. We note that the ℓ-1 norm in (10a) can be replaced by other norms. The optimal value of (10), denoted by ϵ^*, gives the minimum reconstruction error we have to tolerate due to measurement errors.

We then incorporate the objective of minimizing the total edge weight:

$$\min \sum_{A\in\mathcal{A}} w_A \quad (11a)$$

$$\text{s.t. } \sum_{i\in[n]} |\sum_{A:i\in A} w_A - \rho_i| + \sum_{i,j\in[n]} |\sum_{A:\{i,j\}\subseteq A} w_A - \rho_{ij}| \leq \epsilon, \quad (11b)$$

$$w_A \geq 0, \qquad \forall A \in \mathcal{A}. \quad (11c)$$

This optimization tries to minimize the total weight (11a) subject to the constraints of approximately satisfying the measurements (11b) and ensuring nonnegativity. The parameter ϵ is used to trade off the reconstruction error and the total weight of the inferred topology. At the minimum, it should account for measurement errors, i.e., $\epsilon \geq \epsilon^*$. As in (10), other norms can be used instead of the ℓ-1 norm in (11b). Problem (11) is again a convex optimization. In the special case of $\epsilon = 0$, (11b) can be replaced by linear constraints (9), and (11) becomes a linear program. After obtaining the solution $(w_A)_{A\in\mathcal{A}}$, we can use any topology construction algorithm, such as Algorithm 1 presented later, to construct a minimum weight representation (with up to ϵ reconstruction error).

Remark: In cases that the distribution $g(\cdot)$ of measurement errors is known, we can incorporate this information by performing the *maximum likelihood estimation (MLE)* of the category weights. This is a constrained optimization similar to (10), with the objective (10a) replaced by $\max g((\sum_{A:i\in A} w_A - \rho_i)_{i\in[n]}, (\sum_{A:\{i,j\}\subseteq A} w_A - \rho_{ij})_{i,j\in[n]})$. Similarly, we can compute a minimum weight representation by solving a variation of (11), with (11b) replaced by a constraint of the form $g(\cdot) \geq \delta$, where δ is no greater than the maximum likelihood.

2) Topology Construction: Given the inferred category weights $(w_A)_{A\in\mathcal{A}}$, we want to find a topology and paths in the topology such that the sum of edge weights in each category matches the given value. As an empty category (i.e., containing no edge) must have a zero weight and a nonempty category (i.e., containing at least one edge) can be assigned any weight, it suffices to *represent* each positive-weight category by at least one edge, i.e., if $w_A > 0$, there must exist at least one edge in Γ_A. Generally, there are multiple topologies that can represent a given set of positive-weight categories, and our objective is to find the "simplest" topology in the following sense.

Definition 2. Given category weights $(w_A)_{A\in\mathcal{A}}$, the *topology construction problem* aims to construct a directed graph $\mathcal{G} = (V, E)$ with the minimum number of vertices (*minimum order representation*) or edges (*minimum size representation*), together with n paths in $\mathcal{G}$, such that all the positive-weight categories are represented.

Algorithm 1: Clique Embedding (CE)

input : Number of measurement flows n and category weights $(w_A)_{A\in\mathcal{A}}$
output: Inferred topology $\mathcal{G}$ and flow paths $\{p_i\}_{i=1}^n$
1 find the minimum directed clique $\mathcal{C}$ with at least $|\{A \in \mathcal{A} : w_A > 0\}|$ edges;
2 **foreach** $A \in \mathcal{A}$ *such that* $w_A > 0$ **do**
3 randomly select an unselected edge in $\mathcal{C}$, and assign it to category Γ_A and weight w_A;
4 create a new vertex r;
5 **foreach** $i = 1, \ldots, n$ **do**
6 find continuous edge sequences formed by edges assigned to categories $\{\Gamma_A : A \in \mathcal{A}, i \in A\}$: $p_{i,1}, \ldots, p_{i,m_i}$;
7 **foreach** *edge sequence* $p_{i,j}$ **do**
8 create an edge from r to the beginning of $p_{i,j}$ and an edge from the end of $p_{i,j}$ to r, both of zero weight;
9 p_i is the concatenation of the cycles formed by going from r to $p_{i,j}$ and back to r for $j = 1, \ldots, m_i$;
10 $\mathcal{G}$ consists of all the selected edges in $\mathcal{C}$, vertex r, and all the edges between r and $\mathcal{C}$;

This definition decouples the weight inference problem and the topology construction problem such that we can "mix" solutions to these problems. For example, an algorithm for constructing a minimum order/size representation from given category weights can take the output of any weight inference algorithm, although intuitively the algorithm that gives the sparsest solution (i.e., minimizing the number of positive category weights) helps to minimize the order/size of the constructed topology. In this sense, (11) is an ℓ-1 approximation to the weight inference solution that facilitates the construction of the overall minimum order/size representation.

Although the optimal solution for the minimum order representation needs not be the same as the optimal solution for the minimum size representation, we are able to develop an algorithm that is near-optimal for both types of representations.

Algorithm: Our idea is to "embed" edges representing the categories with non-zero weights into the minimum directed clique (i.e., complete directed graph) with sufficiently many edges. This is because we must construct at least one edge for each category with non-zero weight, and the clique is the smallest graph that contains a given number of edges. This is the initial idea behind our topology construction algorithm, referred to as *Clique Embedding (CE)*, shown in Algorithm 1 (lines 1–3).

However, the embedded edges may not form valid paths, i.e., for a given $i \in [n]$, the embedded edges in categories $\{\Gamma_A : A \in \mathcal{A}, i \in A\}$ may not form a sequence of pairwise adjacent edges. To generate valid paths, we construct a special vertex r (line 4), which is connected to/from each continuous sequence of embedded edges that need to be traversed by p_i (lines 7–8). Thus, we can "stitch together" the edge sequences via r to form a valid path (line 9). Fig. 5 illustrates the idea: if the embedding generates three continuous edge sequences $p_{i,1}$, $p_{i,2}$, and $p_{i,3}$ for some $i \in [n]$, then the constructed path p_i goes from r to $p_{i,1}$ and back to r, then to $p_{i,2}$ and back to r, and finally to $p_{i,3}$ and back to r (ordering does not matter).

Given the set E_i of embedded edges that need to be traversed by p_i (i.e., assigned to categories $\{\Gamma_A : A \in \mathcal{A}, i \in A\}$), we find the continuous edge sequences (line 6) as follows: (i) initialize each edge sequence $p_{i,j}$ as a one-hop sequence containing a randomly selected edge in E_i that has not been

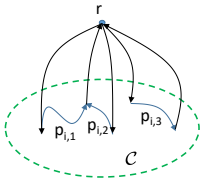


Fig. 5. Using r and its adjacent edges to stitch the embedded edge sequences $p_{i,1}$, $p_{i,2}$, and $p_{i,3}$ into a valid path p_i.

covered by the existing edge sequences;
(ii) extend $p_{i,j}$ by adding one edge at a time from the uncovered edges in E_i, until no more extension can be made;
(iii) if there are still uncovered edges in E_i, repeat (i–ii).

Performance: Among all the feasible topologies, Algorithm 1 gives a near-optimal representation of the ground truth topology in the following sense.

Theorem III.4. The topology $\mathcal{G}$ given by Algorithm 1 is

(a) near-optimal in minimizing the order, in that $\mathcal{G}$ has at most one more vertex than the minimum order representation, and

(b) asymptotically near-optimal in minimizing the size, in that for any $\epsilon > 0$, the number of edges in $\mathcal{G}$ is no more than $(1+\epsilon)$ times the number of edges in the minimum size representation for all sufficiently large $|\{A \in \mathcal{A} : w_A > 0\}|$.

Proof. Let $k_e := |\{A \in \mathcal{A} : w_A > 0\}|$ and $h(k_e) := \min\{m : m(m-1) \geq k_e\}$ be the number of vertices in the minimum clique with at least k_e edges.

First, the minimum order representation needs at least one edge in each positive-weight category, and hence its number of vertices is at least $h(k_e)$. The topology given by Algorithm 1 contains $h(k_e)+1$ vertices. Hence, claim (a) holds.

Moreover, Algorithm 1 constructs at most $k_e + 2h(k_e)$ edges, as there are at most $2h(k_e)$ edges between r and vertices in the clique. The minimum size representation has at least k_e edges. The approximation ratio is thus upper-bounded by $1 + 2h(k_e)/k_e$. As $h(k_e) = O(\sqrt{k_e})$, for every $\epsilon > 0$, $\exists k_0$ such that $2h(k_e)/k_e \leq \epsilon$ for all $k_e \geq k_0$, proving claim (b). □

Example: Consider the input of $n = 3$ and $w_A > 0$ for all $A \in \mathcal{A}$. Fig. 6 illustrates two possible outcomes of Algorithm 1, together with the set of embedded edges that need to be traversed by each path. In case Fig. 6 (a), we see that there is actually no need to add vertex r, i.e., $\mathcal{G} - r$ is still a feasible solution, as the embedded edges for each i already form a valid path. In case Fig. 6 (b), however, the embedded edges do not form valid paths (E_1 contains 2 edge sequences, E_2 contains 3 sequences, and E_3 contains 2 sequences), and thus r and its neighboring edges are needed. As any feasible topology contains at least 4 vertices and 7 edges, Algorithm 1 provides a solution that is minimum in both order and size in case (a) (after removing r), but not in case (b). We note that it may be possible to strategically embed edges and assign their categories to minimize the order/size of the output, which we leave to future work.

IV. Solutions based on Path Length and Service Information

We now revisit the problem when information about the services required by each flow is also used for inference, including the source s_i, the destination t_i, and the service chain $\mathbf{c}_i$ ($i \in [n]$).

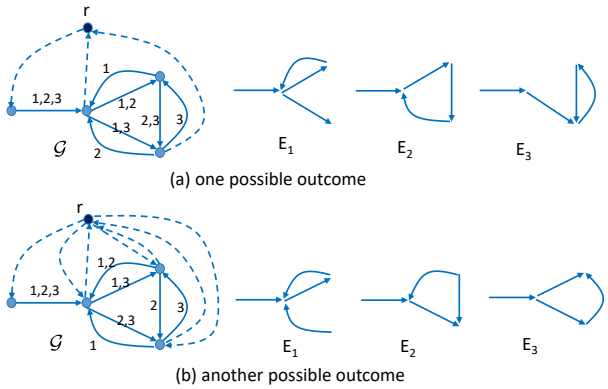


Fig. 6. Two possible outcomes of Algorithm 1 for $n = 3$. Solid line: embedded edges; dashed line: edges to/from r; E_i: embedded edges that need to be traversed by p_i; edge label: the category index.

While the service information distinguishes our problem from all the existing topology inference problems, we can still reuse some of previous solutions. Specifically, as the service information does not inform about the edge weights, we can still divide the problem into two subproblems: (1) *weight inference*, and (2) *VNF topology construction*. Subproblem (1) has the same input and output as in Section III-B1, and hence results therein apply. Subproblem (2) takes both the inferred category weights and the service information as input, and outputs a directed, vertex-labeled and edge-weighted graph $\mathcal{G} = (V, E, l, w)$ that represents the VNF overlay topology. The focus here is subproblem (2). The challenge in VNF topology construction is to preserve the service chains while constructing at least one edge in each positive-weight category.

A. Existence of Single-copy Representation

While the ground truth topology may contain multiple instances of the same VNF, we show that it is always possible to construct an equivalent topology that contains at most one instance per VNF, referred to as a *single-copy representation*.

Theorem IV.1. For each VNF topology $\mathcal{G}$, there exists an equivalent single-copy representation $\widetilde{\mathcal{G}}$, i.e., each $f_i \in \mathcal{F}$ is assigned to at most one vertex in $\widetilde{\mathcal{G}}$.

Proof. We prove by construction. Consider an arbitrary VNF topology $\mathcal{G}$. Let $\mathcal{N}^-(v)$ and $\mathcal{N}^+(v)$ denote the incoming/outgoing neighbors of vertex v, i.e., vertices with edges to/from v. For every two vertices labeled by the same (non-dummy) VNF f_i, denoted by f_i^1 and f_i^2, we have four cases: (1) $\mathcal{N}^-(f_i^1) \cap \mathcal{N}^-(f_i^2) = \emptyset$ and $\mathcal{N}^+(f_i^1) \cap \mathcal{N}^+(f_i^2) = \emptyset$, (2) $\mathcal{N}^-(f_i^1) \cap \mathcal{N}^-(f_i^2) \neq \emptyset$ and $\mathcal{N}^+(f_i^1) \cap \mathcal{N}^+(f_i^2) = \emptyset$, (3) $\mathcal{N}^-(f_i^1) \cap \mathcal{N}^-(f_i^2) = \emptyset$ and $\mathcal{N}^+(f_i^1) \cap \mathcal{N}^+(f_i^2) \neq \emptyset$, and (4) $\mathcal{N}^-(f_i^1) \cap \mathcal{N}^-(f_i^2) \neq \emptyset$ and $\mathcal{N}^+(f_i^1) \cap \mathcal{N}^+(f_i^2) \neq \emptyset$. We "merge" f_i^1 and f_i^2 as in Fig. 7: in case (1), we directly merge them; in case (2), we replace f_i^1 by a dummy denoted by f_0^1, which is connected to f_i^2, and rewire outgoing edges of f_i^1 to start from f_i^2; in case (3), we replace f_i^1 by a dummy f_0^1, connected from f_i^2, and rewire incoming edges of f_i^1 to end at f_i^2; in case (4), we replace f_i^1 by a dummy f_0^1, connected to/from f_i^2. Each path traversing f_i^1 will traverse (f_0^1, f_i^2) in case (2), (f_i^2, f_0^1) in case (3), and (f_0^1, f_i^2, f_0^1) in case (4). Each merge operation reduces the number of duplicate VNF instances by one, while preserving the service chains and the represented

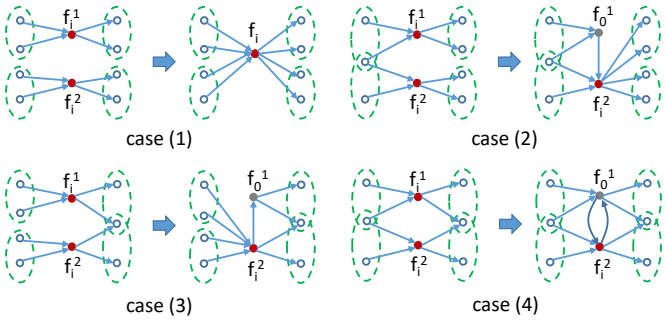


Fig. 7. Merge operation (f_i^1, f_i^2: instances of the same VNF; f_0^1: dummy).

categories. Repeatedly applying this operation will then give a single-copy representation that is equivalent to $\mathcal{G}$. □

Moreover, the simplest single-copy representation is nearly as simple as the overall simplest representation.

Corollary IV.2. a) The minimum order single-copy representation has as few vertices as the minimum order representation.

b) The minimum size single-copy representation has at most $2R$ more edges than the minimum size representation $\mathcal{G}^*$, where R is the number of duplicate VNF instances in $\mathcal{G}^*$.

Proof. As the merge operation defined in the proof of Theorem IV.1 reduces the number of duplicates by one, while creating no extra vertex and at most 2 extra edges, applying it to the minimum order/size representation yields the result. □

B. Construction of Single-copy Representation

In a single-copy representation, we know that there will be a unique vertex corresponding to each source, destination, or type of VNF. However, simply constructing a topology by connecting each pair of vertices that are adjacent in a service chain by an edge may not give a feasible solution, as it may not represent all the positive-weight categories. For example, consider the input in Fig. 8 (a). Service chains p'_1 and p'_2 (including first/last hop) for flows d_1 and d_2 do not have any pair of adjacent vertices in common, but weight inference shows that the paths p_1 and p_2 of these flows share at least one common edge, as the weight of category $\Gamma_{\{1,2\}}$ is positive. This implies that there must be hidden vertices (i.e., branching/joining points) shared by these paths that create shared edges without violating the service chains, as such vertices only "run" dummy VNF. Thus, the VNF topology construction problem reduces to: augment the union of service chains with dummies such that all the positive-weight categories will be represented in the augmented graph. To this end, we first give a feasible solution that is generally suboptimal in order/size, and then present an optimization-based approach to construct the minimum order/size single-copy representation.

1) Extension of Clique Embedding: In the supplementary file, we have shown an algorithm *Clique Embedding++ (CE++)* that can build a feasible solution by extending the previous topology construction algorithm (Algorithm 1) to incorporate service chains. The inferred topology, however, is generally suboptimal in that it may contain more vertices/edges than necessary. To find the simplest solution in terms of the minimum order/size single-copy representation, we introduce a clean-slate solution below.

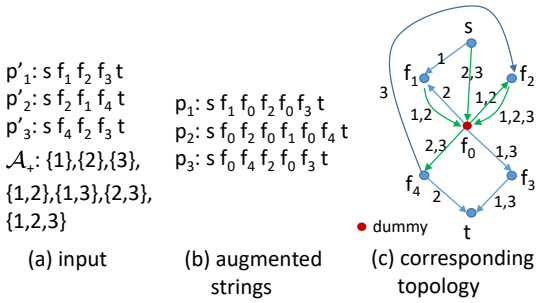


Fig. 8. Example of string augmentation (edge label denotes category index).

2) String Augmentation Problem (SAP): Let $\mathcal{A}_+$ be the set of indices of categories with positive weights, and $p'_i := s_i \oplus \mathbf{c}_i \oplus t_i$ ($\oplus$: concatenation) be the extended service chain of flow d_i, including the endpoints. Viewing each path as a string of vertices, we can formulate the problem of constructing single-copy representations as the following combinatorial optimization problem.

Definition 3 (SAP)**.** Given a set $\mathcal{A}_+$ of subsets of $[n]$ and a set of n initial strings $\{p'_i\}_{i\in[n]}$, the *string augmentation problem (SAP)* aims to augment $(p'_i)_{i\in[n]}$ by inserting dummy letters $f_0^1, f_0^2, \ldots$ (each can be inserted multiple times) such that a certain objective is optimized subject to the constraint that every set in $\mathcal{A}_+$ is represented, i.e., for every $A \in \mathcal{A}_+$, $\exists$ a pair of letters (f_1, f_2) which appear consecutively in string i ($i \in [n]$) if and only if $i \in A$.

SAP with the *minimum order objective* aims at minimizing the number of distinct dummy letters. SAP with the *minimum size objective* aims at minimizing the number of distinct pairs of consecutive letters in all the augmented strings.

Example: Fig. 8 gives an example of the input/output of SAP. The augmented strings provide a VNF topology, where each letter corresponds to a vertex and each string to a path.

3) SAP as Integer Linear Programs (ILPs): We can formulate SAP as ILPs with a polynomial number of variables and constraints. Our formulation assumes that *the service chains are cycle-free* (i.e., no duplicate letters in p'_i). Let $m_{\max}$ be an upper bound on the number of dummy letters and $l_{\max}$ an upper bound on the length of each string. Let $\mathcal{B} := \{s_i, t_i\}_{i\in[n]} \cup \mathcal{F} \cup \{f_0^k\}_{k\in[m_{\max}]}$ denote the set of all the letters. From CE++ (see the supplementary file), we know that $m_{\max} = O(n)$ and $l_{\max} = O(|\mathcal{F}| + n^2)$.

Variables: We use variable $x_{i,j}^f \in \{0,1\}$ to denote if the j-th letter in string i is f. Moreover, we use variable $\delta_k \in \{0,1\}$ to indicate if dummy f_0^k is used in any string, and $\delta_{f_1,f_2} \in \{0,1\}$ to indicate if (f_1, f_2) is used (consecutively) in any string.

Constraints: The first constraint is that there is at most one letter in each position of each string:

$$\sum_{f\in\mathcal{B}} x_{i,j}^f \le 1, \qquad \forall i \in [n],\ j \in [l_{\max}], \tag{12}$$

and the first (or last) letter must correspond to the source (or destination) of the flow:

$$x_{i,1}^{s_i} = 1, \quad x_{i,l_{\max}}^{t_i} = 1, \qquad \forall i \in [n]. \tag{13}$$

We allow $\sum_{f\in\mathcal{B}} x_{i,j}^f = 0$ in (12) to denote that there might be no letter in a position (and hence the augmented string can be shorter than $l_{\max}$). The second constraint is that service chains must be preserved:

$$\sum_{1\le j_1<j_2\le l_{\max}} x^{f_1}_{i,j_1}\cdot x^{f_2}_{i,j_2}=1, \qquad \forall i\in[n],\ (f_1,f_2)\in p'_i, \tag{14}$$

$$\sum_{j\in[l_{\max}]} x^{f}_{i,j}=\mathbf{1}(f\in p'_i), \qquad \forall i\in[n],\ f\in\{s_i,t_i\}_{i\in[n]}\cup\mathcal{F}, \tag{15}$$

which includes preserving the set of non-dummy letters (15) and the order of them (14). Here $\mathbf{1}(\cdot)$ is the indicator function. The third constraint is that each positive-weight category must be represented:

$$\sum_{f_1,f_2\in\mathcal{B}}\prod_{i\in A}\mathbf{1}\Big(\sum_{j=1}^{l_{\max}-1}x^{f_1}_{i,j}x^{f_2}_{i,j+1}>0\Big)\cdot\prod_{i\notin A}\mathbf{1}\Big(\sum_{j=1}^{l_{\max}-1}x^{f_1}_{i,j}x^{f_2}_{i,j+1}=0\Big)>0, \qquad \forall A\in\mathcal{A}_+, \tag{16}$$

where $\sum_{j=1}^{l_{\max}-1}x^{f_1}_{i,j}x^{f_2}_{i,j+1}$ is the number of times (f_1,f_2) appears consecutively in string i. Additionally, for minimum order objective, we need

$$x^{f^k_0}_{i,j}\le\delta_k, \qquad \forall k\in[m_{\max}], i\in[n], j\in[l_{\max}], \tag{17}$$

and for minimum size objective, we need

$$\mathbf{1}\Big(\sum_{j=1}^{l_{\max}-1}x^{f_1}_{i,j}x^{f_2}_{i,j+1}>0\Big)\le\delta_{f_1,f_2}, \qquad \forall f_1,f_2\in\mathcal{B}\text{ and }i\in[n]. \tag{18}$$

Objective: For minimum order objective, the goal is to:

$$\min \sum_{k=1}^{m_{\max}}\delta_k, \tag{19a}$$

$$\text{s.t. (12–17).} \tag{19b}$$

For minimum size objective, the goal is to:

$$\min \sum_{f_1,f_2\in\mathcal{B}}\delta_{f1,f2}, \tag{20a}$$

$$\text{s.t. (12–16) and (18).} \tag{20b}$$

Linearization: Constraints (14,16,18) are non-linear. To linearize them, we introduce the following dependent variables, all in $\{0,1\}$. Variable $\gamma^i_{f_1,f_2,j_1,j_2}$ s.t.

$$\gamma^i_{f_1,f_2,j_1,j_2}\le x^{f_1}_{i,j_1}, \tag{21a}$$

$$\gamma^i_{f_1,f_2,j_1,j_2}\le x^{f_2}_{i,j_2}, \tag{21b}$$

$$\gamma^i_{f_1,f_2,j_1,j_2}\ge x^{f_1}_{i,j_1}+x^{f_2}_{i,j_2}-1 \tag{21c}$$

replaces $x^{f_1}_{i,j_1}\cdot x^{f_2}_{i,j_2}$. Variable $\zeta^i_{f_1,f_2}$ s.t.

$$\sum_{j=1}^{l_{\max}-1}\gamma^i_{f_1,f_2,j,j+1}\le l_{\max}\zeta^i_{f_1,f_2}, \tag{22a}$$

$$\sum_{j=1}^{l_{\max}-1}\gamma^i_{f_1,f_2,j,j+1}\ge\zeta^i_{f_1,f_2} \tag{22b}$$

replaces $\mathbf{1}(\sum_{j=1}^{l_{\max}-1}x^{f_1}_{i,j}x^{f_2}_{i,j+1}>0)$. Variable $\xi^A_{f_1,f_2}$ s.t.

$$\xi^A_{f_1,f_2}\le\zeta^i_{f_1,f_2}, \qquad \forall i\in A, \tag{23a}$$

$$\xi^A_{f_1,f_2}\le 1-\zeta^i_{f_1,f_2}, \qquad \forall i\notin A, \tag{23b}$$

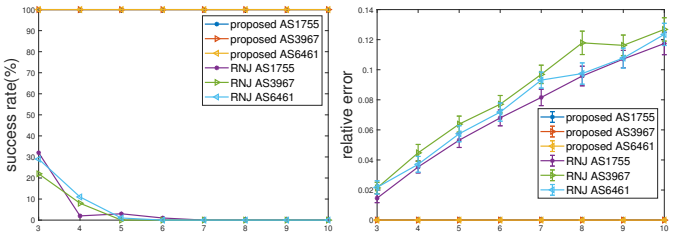
$$\xi^A_{f_1,f_2}\ge\sum_{i\in A}\zeta^i_{f_1,f_2}+\sum_{i\notin A}(1-\zeta^i_{f_1,f_2})-n+1 \tag{23c}$$

replaces $\prod_{i\in A}\mathbf{1}(\sum_{j=1}^{l_{\max}-1}x^{f_1}_{i,j}x^{f_2}_{i,j+1}>0)\cdot\prod_{i\notin A}\mathbf{1}(\sum_{j=1}^{l_{\max}-1}x^{f_1}_{i,j}x^{f_2}_{i,j+1}=0)$.

Using these variables, we can rewrite (14,16,18) as

TABLE I
PARAMETERS OF AS TOPOLOGIES

AS	ISP	#nodes	#links
1755	Ebone (Europe)	172	381
6461	Abovenet (US)	182	294
3967	Exodus (US)	201	434



(a) reconstruction success rate (b) reconstruction error

Fig. 9. Accuracy of reconstructing path/shared path lengths in single-source probing ($|\mathcal{F}|=5$, $S=5$).

$$\sum_{1\le j_1<j_2\le l_{\max}}\gamma^i_{f_1,f_2,j_1,j_2}=1, \qquad \forall i\in[n],\ (f_1,f_2)\in p'_i, \tag{24}$$

$$\sum_{f_1,f_2\in\mathcal{B}}\xi^A_{f_1,f_2}>0, \qquad \forall A\in\mathcal{A}_+, \tag{25}$$

$$\zeta^i_{f_1,f_2}\le\delta_{f_1,f_2}, \qquad \forall f_1,f_2\in\mathcal{B}\text{ and }i\in[n], \tag{26}$$

which converts the problem of constructing the minimum order/size single-copy representation into ILPs.

Complexity: The number of variables in these ILPs is $O(|\mathcal{B}|^2(nl^2_{\max}+|\mathcal{A}_+|))$, and the number of constraints is $O(n|\mathcal{B}|^2(l^2_{\max}+|\mathcal{A}_+|))$. As $|\mathcal{B}|=O(n+|\mathcal{F}|)$, $l_{\max}=O(|\mathcal{F}|+n^2)$, and $|\mathcal{A}_+|=O(n^2)$, both numbers are in $O(n(n+|\mathcal{F}|)^2(n^2+|\mathcal{F}|)^2)$. This implies that the heuristic of LP relaxation with rounding will have a polynomial complexity. We conjecture that solving SAP optimally is NP-hard.

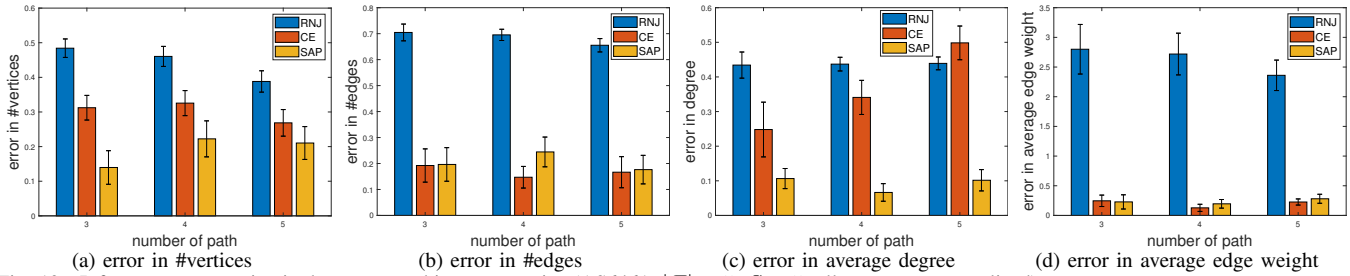
Discussion: The above formulation can be extended to incorporate additional constraints, such as communication capacities of links and processing capacities of servers hosting VNFs. When available, adding these constraints can bring the inferred topology closer to the ground truth, potentially at the cost of a higher computational complexity. We leave the detailed investigation of these extensions to future work.

V. PERFORMANCE EVALUATION

Setting: We evaluate our solutions via data-driven simulations. Due to the lack of public NFV datasets, we synthesize VNF overlay topologies based on real Internet topologies. To this end, we use Rocketfuel *Autonomous System (AS)* topologies [42], which represent IP-level connections between routers in ASs of several ISPs. Parameters of the considered topologies are given in Table I. We assign to each link a weight uniformly distributed in $[0.02,0.1]$, which corresponds to a success rate (or no-queueing probability) of $[0.90,0.98]$ according to the definition of loss-based (or utilization-based) weight in Section II-C.

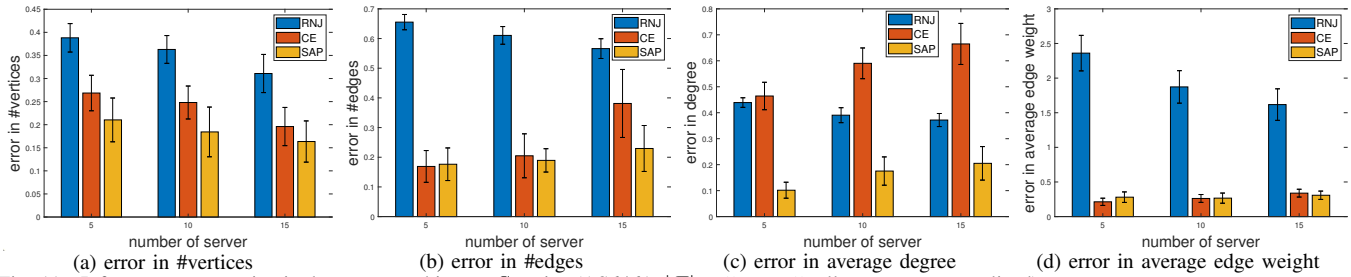
Treating these topologies as the substrate, we generate VNF overlays by randomly selecting S of the high-degree nodes (degree ≥ 9) as servers[2], and randomly placing $|\mathcal{F}|$ VNFs at these servers (one instance per server) while ensuring at least

[2]We collapse a server and its associated router into one node.



(a) error in #vertices (b) error in #edges (c) error in average degree (d) error in average edge weight

Fig. 10. Inference accuracy in single-source probing as n varies (AS6461, $|\mathcal{F}| = 5$, $S = 5$, all errors are normalized).



(a) error in #vertices (b) error in #edges (c) error in average degree (d) error in average edge weight

Fig. 11. Inference accuracy in single-source probing as S varies (AS6461, $|\mathcal{F}| = 5$, $n = 5$, all errors are normalized).

one instance per VNF. We then randomly select $k + m$ of the low-degree nodes (degree ≤ 2) as endpoints, where k are designated as sources and the rest as destinations. This gives us $n = km$ source-destination pairs. For each source-destination pair, we send a flow with a service chain that is a random permutation of $|\mathbf{c}_i|$ different VNFs, where $|\mathbf{c}_i|$ is uniformly distributed in $\{1, \ldots, |\mathcal{F}|\}$. The routing path of each flow is a concatenation of the shortest (hop count) paths from the source to the nearest instance of the first VNF, and then to the nearest instance of the second VNF, etc. All results are averaged over 20 Monte Carlo runs unless stated otherwise, and whenever applicable, we show the confidence interval as "mean $\pm$ standard deviation". We use "node/link" to refer to elements in the substrate, and "vertex/edge" to refer to elements in the overlay. Code and data of our simulations are available at [43].

A. Evaluation of Single-source Probing

Benchmark: As all the existing topology inference algorithms assume that the routing topology for each source is a tree, we use *Rooted Neighbor-Joining (RNJ)* [23] as the benchmark. For single-source probing, RNJ guarantees correct reconstruction if the ground truth topology is a tree.

Reconstruction Accuracy: First, we compare the accuracy of the proposed solution (by solving (11) with $\epsilon = 0$) against RNJ in reconstructing the measured path lengths and shared path lengths, where we assume accurate measurements for both algorithms. Fig. 9 (a) shows the success rate, defined as the fraction of time that all the lengths are reconstructed correctly. Fig. 9 (b) shows the normalized reconstruction error, defined as $\|\hat{\boldsymbol{\rho}} - \boldsymbol{\rho}\|_1 / \|\boldsymbol{\rho}\|_1$, where $\boldsymbol{\rho}$ is the vector of given path/shared path lengths, and $\hat{\boldsymbol{\rho}}$ the reconstructed values. Both are computed over 100 Monte Carlo runs. Clearly, RNJ fails to match the measurements (its success rate tends to zero). This is because it is designed for tree topologies, while our ground truth topologies are no longer trees due to VNF traversals. Meanwhile, our solution is always accurate. This highlights the need to consider general graphs in VNF topology inference. We note that the reconstruction errors of RNJ are only associated with the shared path lengths.

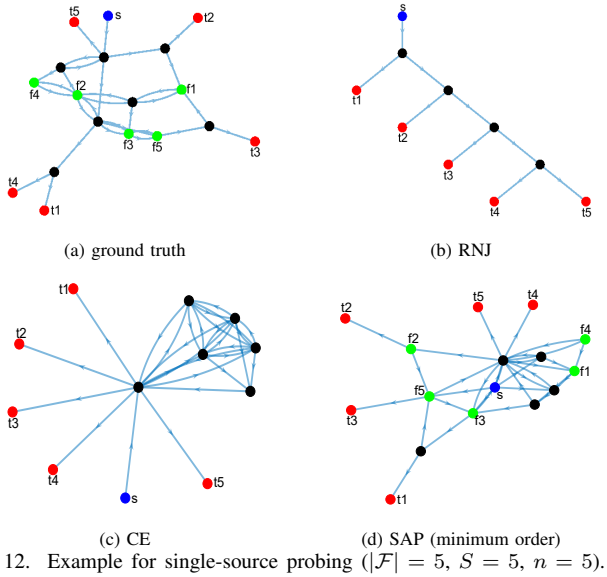
Inference Accuracy: Next, we compare RNJ, our solution without service information—CE (Algorithm 1), and our solution with service information—SAP (Section IV-B3), in terms of the accuracy of the inferred topology. We use the minimum order objective for SAP, solved by CPLEX. We skip CE++ as it performs worse than SAP. To measure the accuracy, we evaluate the normalized error in reconstructing several graph properties, including #vertices, #edges, average vertex degree (including in- and out-degree), and average edge weight.

Fig. 10 shows the result when varying the number of paths n. RNJ incurs substantial error, significantly underestimating the complexity of the ground truth topology by only constructing trees. Meanwhile, CE tends to give overly dense topologies as it aims at embedding edge weights into the smallest graph. By jointly considering path length and service chain information, SAP achieves the best accuracy. Similar observations hold for other AS topologies. Note that although CE better approximates the ground truth in terms of the number of edges (Fig. 10 (b)), it fails to approximate the structure of the ground truth as shown in the detailed example later (see Fig. 12).

We have similar comparisons when varying the number of servers S as in Fig. 11, where some VNFs will have multiple replicas when $S > |\mathcal{F}|$. This result shows that although SAP only constructs single-copy representations, its accuracy is not sensitive to the replication of VNFs.

Detailed Example: To give a concrete idea of how different algorithms perform in resembling the structure of the ground truth topology, we plot a specific realization of the ground truth and the inferred topologies in Fig. 12. In this example, there are 5 VNFs $\{f_1, \ldots, f_5\}$ with one instance each, and we measure 5 paths from source s to destinations $t_1, \ldots, t_5$, with service chains: $\mathbf{c}_1 = (f_3)$, $\mathbf{c}_2 = (f_5, f_3, f_4, f_1, f_2)$, $\mathbf{c}_3 = (f_4, f_1, f_3, f_2, f_5)$, $\mathbf{c}_4 = (f_1, f_3)$, and $\mathbf{c}_5 = (f_3, f_5, f_1, f_4)$.

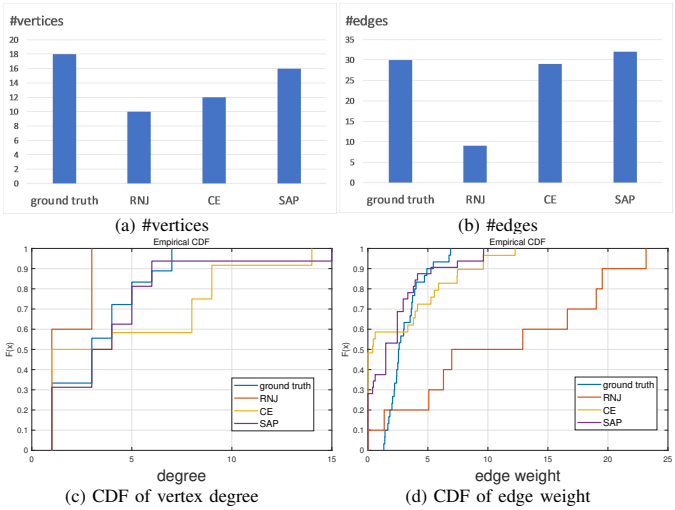
Intuitively, only SAP resembles the structure of the ground truth topology, where the source and the destinations are connected via a densely-connected core that hosts the VNFs.



(a) ground truth (b) RNJ

(c) CE (d) SAP (minimum order)

Fig. 12. Example for single-source probing ($|\mathcal{F}| = 5$, $S = 5$, $n = 5$). ●: source; ●: destination; ●: VNF; ●: dummy.



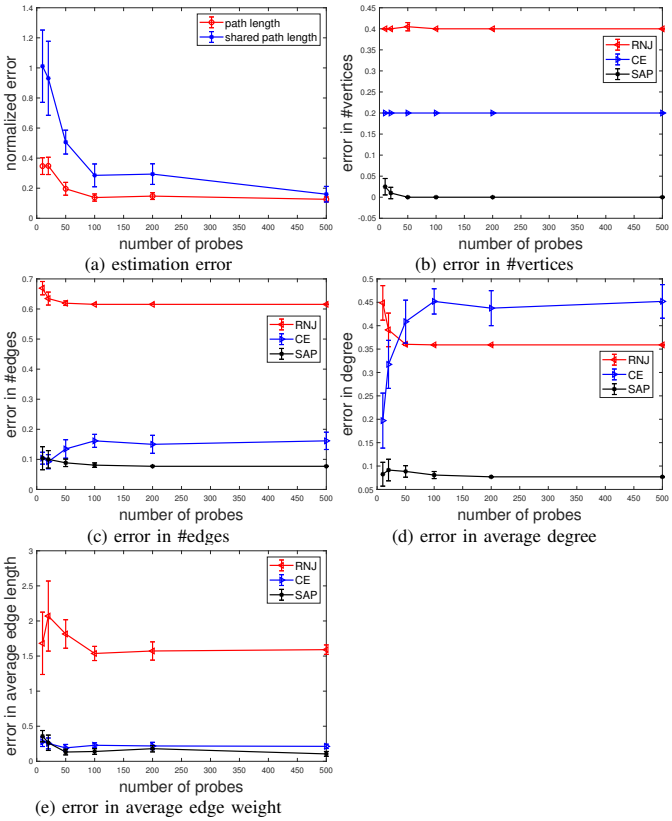
(a) #vertices (b) #edges

(c) CDF of vertex degree (d) CDF of edge weight

Fig. 13. Detailed metrics of the example in Fig. 12.

Detailed metrics from the example are shown in Fig. 13. Overall, SAP outperforms the other algorithms with respect to both the graph structure and the metrics.

Impact of Measurement Error: So far we have assumed accurate measurements of path/shared path lengths. To evaluate the impact of the error in measuring these lengths, we perform a packet-level simulation. An edge with weight w_e is configured to drop packets with probability $1-e^{-w_e}$. We send N pairs of probing packets on each pair of paths to estimate the path/shared path lengths as described in Section II-C1 and ignore negative values. Fig. 14 (a) shows the normalized error in estimating the path/shared path lengths, and Fig. 14 (b-e) show the corresponding topology inference accuracy in terms of various graph properties. We see that all the topology inference algorithms converge quickly (at $N \approx 100$), even when there are still substantial errors in the length measurements (15% for path lengths and 50% for shared path lengths). This is because these algorithms only depend on coarse statistics of the measurements, e.g., RNJ only depends on the ranking of certain functions of the measurements, and our algorithms (CE and SAP) only depend on the set of positive-weight categories



(a) estimation error (b) error in #vertices

(c) error in #edges (d) error in average degree

(e) error in average edge weight

Fig. 14. Packet-level simulation for single-source probing (AS6461, $|\mathcal{F}| = 5$, $S = 5$, $n = 3$).

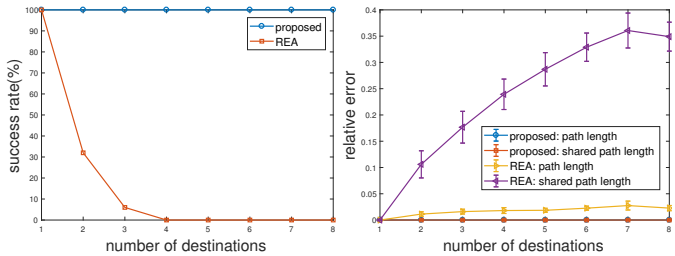
obtained by applying weight inference to the measurements. We have also evaluated the impact of measurement error on other metrics such as the number of edges, the average degree, and the average edge weight, and the results are similar.

Note: The parameters and the number of Monte Carlo runs in our evaluation are limited by the high complexity of solving the weight inference problem (Section III-B1) and SAP (Section IV-B3). We conjecture that both problems are NP-hard, and leave the proof of hardness and the development of efficient heuristics to future work.

B. Evaluation of Multi-source Probing

Benchmark: We use *Receiver Elimination Algorithm (REA)* as the benchmark [30]. REA can correctly reconstruct the 2-by-N topology (2 sources, N destinations), under the assumption that routing from each source follows a tree. It is worth mentioning that the original version of REA needs to query *quartets*, but we have adapted it to take as input the path lengths and the shared path lengths. As REA is designed for two-source probing, we fix the number of sources as 2 in the sequel, although our solution does not make this assumption.

Reconstruction Accuracy: Similar to the case of single-source probing, we compare the accuracy of our solution against REA in reconstructing the measured path and shared path lengths, assuming accurate measurements for both algorithms. Fig. 15 (a) shows the success rate and Fig. 15 (b) shows the normalized reconstructed error, as the number of destinations increases. Both plots are computed over 100 Monte Carlo runs. We see that when the routing is no longer along trees, REA cannot guarantee that the inferred topology will be



(a) reconstruction success rate (b) reconstruction error

Fig. 15. Accuracy of reconstructing path/shared path lengths in multi-source probing ($|\mathcal{F}| = 5$, $S = 5$).

consistent with all the measurements, but our solution can. A difference from the case of single-source probing is that while RNJ only has errors in reconstructing the shared path lengths, REA has errors in reconstructing both the path lengths and the shared path lengths. This result reiterates the importance of considering general graphs in VNF topology inference.

Inference Accuracy: Next, we compare REA, CE, and SAP (with the minimum order objective) in their accuracy of topology inference, measured by the normalized error in #vertices, #edges, average vertex degree, and average edge weight.

Fig. 16 shows the result when varying the number of destinations. Not surprisingly, it gives similar results as in single-source probing: REA only constructs sparse topologies that cannot capture the complexity of the ground truth, and CE only constructs dense topologies that can also be far from the ground truth. Compared with these two algorithms, SAP achieves the best accuracy by considering the service chain.

Detailed Example: To be more specific, we show an example in Fig. 17. There are 5 VNFs $\{f_1, \ldots, f_5\}$ with one instance each, and we measure 6 paths from two sources s_1, s_2 to each of the destinations t_1, t_2, t_3. The service chains (including source/destination) are: $p'_1 = (s_1, f_3, t_1)$, $p'_2 = (s_1, f_3, f_4, t_2)$, $p'_3 = (s_1, f_5, f_4, f_2, f_1, f_3, t_3)$, $p'_4 = (s_2, f_5, f_1, f_2, f_3, f_4, t_1)$, $p'_5 = (s_2, f_1, t_2)$ and $p'_6 = (s_2, f_3, f_4, f_5, f_1, t_3)$.

We see that the topology inferred by SAP best resembles the structure of the ground truth topology. To quantify this resemblance, we plot #vertices, #edges, CDF of vertex degrees, and CDF of edge weights in Fig. 18. Overall, SAP performs the best in these metrics, especially when compared with REA.

Impact of Measurement Error: To evaluate the impact of measurement error, we performed a packet-level simulation for multi-source probing, under the same setting as single-source probing (Fig. 14). Fig. 19 (a) shows the normalized error in estimating the path/shared path lengths, and Fig. 19 (b-e) show the corresponding topology inference accuracy in terms of various graph properties. Similar to Fig. 14, all the topology inference algorithms converge fast (at ≈ 200 probes), when there are still 10% errors in the measured path lengths and 14% errors in the measured shared path lengths. Similar convergence behavior has been observed for other metrics.

VI. Conclusion

We consider, for the first time, the problem of inferring the structure and state of NFV networks based on service chains and end-to-end performance measurements. We show that existing tree-based algorithms cannot guarantee a feasible solution that is consistent with all the measurements, which motivates us to propose a novel two-step solution designed to construct the simplest logical topology that is equivalent to the ground truth. Extensive evaluations show that the proposed solution significantly improves the accuracy over several state-of-the-art topology inference algorithms.

We note that the proposed solution contains nontrivial convex or combinatorial optimization problems. Although not the focus of this work, efficient algorithms for solving these optimizations are certainly of interest for future work.

References

[1] 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.

[2] J. Sherry and S. Ratnasamy, "A survey of enterprise middlebox deployments," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2012-24, Feb 2012. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-24.html

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

[4] "AT&T vision alignment challenge technology survey," AT&T Domain 2.0 Vision White Paper, November 2013. [Online]. Available: https://www.att.com/Common/about_us/pdf/AT&TDomain2.0VisionWhitePaper.pdf

[5] R. Cohen, L. Lewin-Eytan, J. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *IEEE INFOCOM*, April 2015.

[6] H. Moens and F. D. Turck, "VNF-P: a model for efficient placement of virtualized network functions," in *IEEE CNSM*, November 2014.

[7] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *IEEE CloudNet*, October 2014.

[8] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network function placement for nfv chaining in packet/optical datacenters," *Journal of Lightwave Technology*, vol. 33, no. 8, pp. 1565–1570, 2014.

[9] T. Loukovszki and S. Schmid, "Online admission control and embedding of service chains," in *ACM SIROCCO*, July 2015.

[10] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "On orchestrating virtual network functions," in *IEEE CNSM*, November 2015.

[11] M. Barcelo, J. Llorca, A. M. Tulino, and N. Raman, "The cloud service distribution problem in distributed cloud networks," in *IEEE ICC*, June 2015.

[12] T.-W. Kuo, B.-H. Liou, K. C.-J. Lin, and M.-J. Tsai, "Deploying chains of virtual network functions: On the relation between link and server usage," in *IEEE INFOCOM*, April 2016.

[13] H. Feng, J. Llorca, A. M. Tulino, D. Raz, and A. F. Molisch, "Approximation algorithms for the nfv service distribution problem," in *IEEE INFOCOM*, April 2017.

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

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

[16] 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.

[17] H. Nguyen and R. Zheng, "A binary independent component analysis approach to tree topology inference," *IEEE Transactions on Signal Processing*, vol. 61, no. 12, pp. 3071–3080, June 2013.

[18] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, "Multicast topology inference from measured end-to-end loss," *IEEE Transactions on Information Theory*, vol. 48, no. 1, pp. 26–45, January 2002.

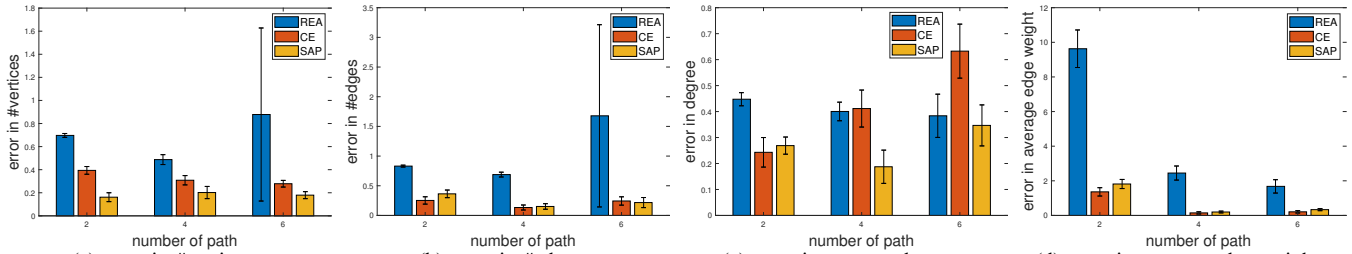
[19] ——, "Multicast topology inference from end-to-end measurements," *Advances in Performance Analysis*, vol. 3, pp. 207–226, 2000.

[20] 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.

[21] N. G. Duffield and F. L. Presti, "Network tomography from measured end-to-end delay covariance," *IEEE/ACM Transactions on Networking*, vol. 12, no. 6, pp. 978–992, December 2004.

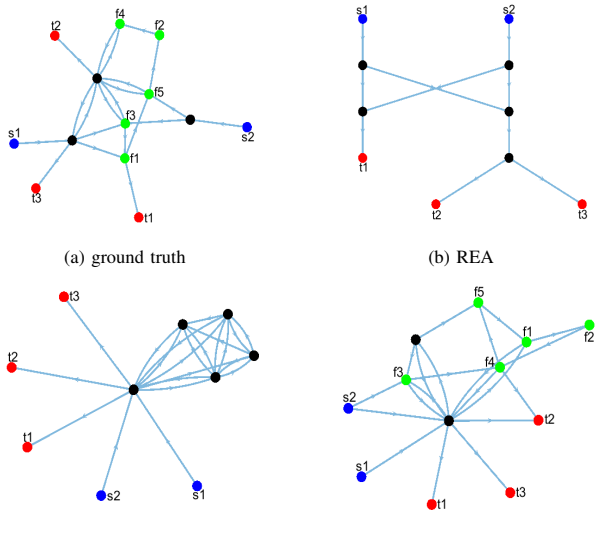
[22] N. G. Duffield, J. Horowitz, and F. L. Presti, "Adaptive multicast topology inference," in *IEEE INFOCOM*, 2001.

[23] 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.



(a) error in #vertices (b) error in #edges (c) error in average degree (d) error in average edge weight

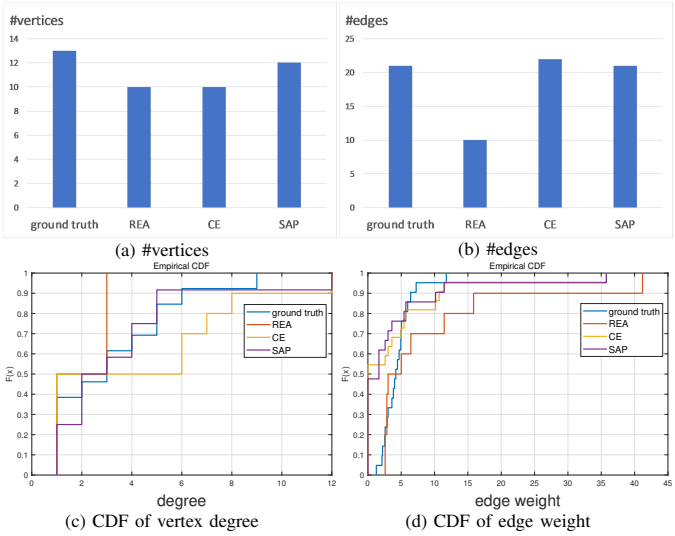
Fig. 16. Inference accuracy in multi-source probing (AS6461, $|\mathcal{F}| = 5$, $S = 5$, all errors are normalized).



(a) ground truth (b) REA

(c) CE (d) SAP (minimum order)

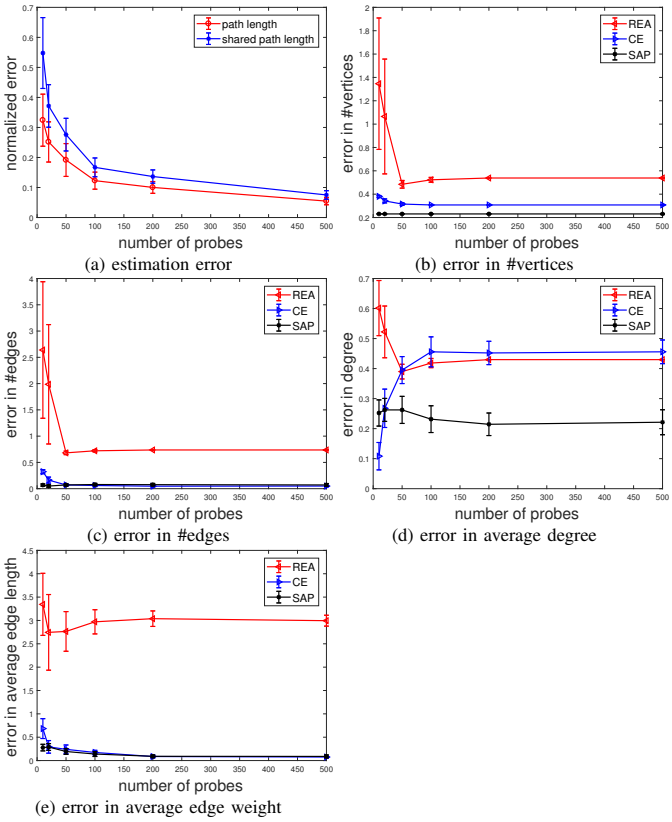
Fig. 17. Example for multi-source probing ($|\mathcal{F}| = 5$, $S = 5$, $n = 6$). ●: source; ●: destination; ●: VNF; ●: dummy.



(a) #vertices (b) #edges

(c) CDF of vertex degree (d) CDF of edge weight

Fig. 18. Detailed metrics of the example in Fig. 17



(a) estimation error (b) error in #vertices

(c) error in #edges (d) error in average degree

(e) error in average edge weight

Fig. 19. Packet-level simulation for multi-source probing (AS6461, $|\mathcal{F}| = 5$, $S = 5$, $n = 4$ (2 sources, 2 destinations)).

[24] J. Ni and S. Tatikonda, "Network tomography based on additive metrics," *IEEE Transactions on Information Theory*, vol. 57, no. 12, pp. 7798–7809, December 2011.

[25] M. Coates, R. Castro, M. Gadhiok, R. King, Y. Tsang, and R. Nowak, "Maximum likelihood network topology identification from edge-based unicast measurements," in *ACM SIGMETRICS*, June 2002.

[26] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1999.

[27] B. Eriksson, P. Barford, and R. Nowak, "Network discovery from passive measurements," in *ACM SIGCOMM*, 2008.

[28] M. Rabbat, R. Nowak, and M. Coates, "Multiple source, multiple destination network tomography," in *IEEE INFOCOM*, 2004.

[29] 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.

[30] 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.

[31] P. Sattari, C. Fragouli, and A. Markopoulou, "Active topology inference using network coding," *Physical Communication*, vol. 6, pp. 142–163, March 2013.

[32] R. Jithin and B. K. Dey, "Exact topology inference for DAGs using network coding," in *IEEE International Symposium on Network Coding (NetCod)*, June 2012.

[33] A. Krishnamurthy and A. Singh, "Robust multi-source network tomography using selective probes," in *IEEE INFOCOM*, March 2012.

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

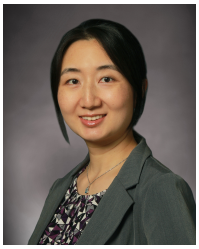
[35] V. Ramasubramanian, D. Malkhi, F. Kuhn, M. Balakrishnan, A. Gupta, and A. Akella, "On the treeness of internet latency and bandwidth," in *ACM SIGMETRICS*, June 2009.

[36] A. Sabnis, R. K. Sitaraman, and D. Towsley, "OCCAM: An optimization based approach to network inference," in *The Workshop on MAthematical performance Modeling and Analysis (MAMA)*, June 2018.

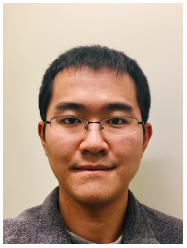
[37] G. Berkolaiko, N. Duffield, M. Ettehad, and K. Manousakis, "Graph Reconstruction from Path Correlation Data," *ArXiv e-prints*, 2018.

[38] N. Duffield, F. L. Presti, V. Paxson, and D. Towsley, "Network loss tomography using striped unicast probes," *IEEE/ACM Transactions on Networking*, vol. 14, no. 4, pp. 697–710, August 2006.
[39] J. Pearl, *Probabilistic Reasoning in Intelligent Systems—Networks of Plausible Inference*. Morgan Kaufmann, 1988.
[40] R. Castro, M. Coates, and R. Nowak, "Likelihood based hierarchical clustering," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2308–2321, August 2004.
[41] A. S. Teixeira, P. T. Monteiro, J. A. Carrico, M. Ramirez, and A. P. Fancisco, "Not seeing the forest for the trees: Size of the minimum spanning trees (MSTs) forest and branch significance in MST-based phylogenetic analysis," *PLoS ONE*, vol. 10, no. 3, March 2015.
[42] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *ACM SIGCOMM*, August 2002.
[43] "NFV-network-topology-inference," GitHub, 2020. [Online]. Available: https://github.com/yileilin/NFV-network-topology-inference

Yilei Lin (S'19) received the B.S. degree in information security from University of Science and Technology of China in 2017 and is a Ph.D. candidate in Computer Science in Pennsylvania State University, advised by Prof. Ting He. Her research interest includes computer networking, statistical inference and queuing network.



Ting He (SM'13) received the B.S. degree in computer science from Peking University, China, in 2003 and the Ph.D. degree in electrical and computer engineering from Cornell University, Ithaca, NY, in 2007. Dr. He is an Associate Professor in the School of Electrical Engineering and Computer Science at Pennsylvania State University, University Park, PA. Between 2007 and 2016, she was a Research Staff Member in the Network Analytics Research Group at the IBM T.J. Watson Research Center, Yorktown Heights, NY. Her work is in the broad areas of computer networking, network modeling and optimization, and statistical inference. Dr. He is an Associate Editor for IEEE Transactions on Communications (2017-2020) and IEEE/ACM Transactions on Networking (2017-2021). She was the Membership co-chair of ACM N2Women in 2013-2014 and was listed in "N2Women: Rising Stars in Networking and Communications" in 2017. She received the Research Division Award and multiple Outstanding Contributor Awards from IBM, the Most Collaboratively Complete Publications Award from ITA, the Best Paper Award at the 2013 International Conference on Distributed Computing Systems (ICDCS), the Outstanding Student Paper Award at the 2015 ACM SIGMETRICS, and the Best Student Paper Award at the 2005 International Conference on Acoustic, Speech and Signal Processing (ICASSP).



Shiqiang Wang (M'15) received his Ph.D. from the Department of Electrical and Electronic Engineering, Imperial College London, United Kingdom, in 2015. Before that, he received his master's and bachelor's degrees at Northeastern University, China, in 2011 and 2009, respectively. He joined IBM T. J. Watson Research Center in 2016 as a Research Staff Member, where he was also a Graduate-level Co-op in the summers of 2014 and 2013. In the fall of 2012, he was at NEC Laboratories Europe, Heidelberg, Germany. His current research focuses on theoretical and practical aspects of mobile edge computing, cloud computing, and machine learning. Dr. Wang currently serves as an associate editor of IEEE Access. He served as a technical program committee (TPC) member of several international conferences including IEEE ICDCS, IJCAI, WWW, IFIP Networking, IEEE GLOBECOM, IEEE ICC, and as a reviewer for a number of international journals and conferences. He received the IBM Outstanding Technical Achievement Award (OTAA) in 2019, multiple Invention Achievement Awards from IBM since 2016, Best Paper Finalist of the IEEE International Conference on Image Processing (ICIP) 2019, and Best Student Paper Award of the Network and Information Sciences International Technology Alliance (NIS-ITA) in 2015.

Kevin Chan (SM'18) received the B.S. degree in electrical and computer engineering and engineering and public policy from Carnegie Mellon University, Pittsburgh, PA, USA, in 2001, and the M.S. and Ph.D. degrees in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2003 and 2008, respectively. He is currently a Research Scientist with the Computational and Information Sciences Directorate, U.S. Army Combat Capabilities Development Command, Army Research Laboratory, Adelphi, MD, USA. He is actively involved in research on network science, distributed analytics, and cybersecurity. He has received multiple best paper awards and the NATO Scientific Achievement Award. He has served on the Technical Program Committee for several international conferences, including IEEE DCOSS, IEEE SECON, and IEEE MILCOM. He is a Co-Editor of the IEEE Communications MagazineMilitary Communications and Networks Series.

Stephen Pasteris gained a BA+MA in Mathematics from Kings College of the University of Cambridge. After completing his BA he then went on to gain a PhD in Computer Science from University College London: his thesis focusing on the development of efficient algorithms for machine learning on networked data. Stephen is now a Research Associate at University College London where he primarily researches online machine learning.

Looking Glass of NFV: Inferring the Structure and State of NFV Network from External Observations — Supplementary File

Yilei Lin, *Student Member, IEEE*, Ting He, *Senior Member, IEEE,* Shiqiang Wang, *Member, IEEE*, Kevin Chan, *Senior Member, IEEE,* and Stephen Pasteris

Algorithm 1: Clique Embedding++ (CE++)

input : Service chains $(p'_i)_{i\in[n]}$ and category weights $(w_A)_{A\in\mathcal{A}}$

output: Inferred VNF topology $\mathcal{G}$ and flow paths $\{p_i\}_{i\in[n]}$

1 $\mathcal{G}_c \leftarrow \bigcup_{i\in[n]} p'_i$;

2 find the set $\{\Gamma_A : A \in \mathcal{A}'\}$ of positive-weight categories not represented in $\mathcal{G}_c$;

3 embed $\mathcal{A}'$ into the minimum clique $\mathcal{C}$ with at least $|\mathcal{A}'|$ edges as in lines 1–3 of Algorithm 1;

4 **foreach** $e \in E_0$ **do**

5 create a vertex u_e, and replace edge e by edges $(s(e), u_e)$ and $(u_e, t(e))$;

6 **foreach** $i = 1, \ldots, n$ **do**

7 connect u_{e_i} to/from the beginning/end of each edge sequence in $\mathcal{C}$ formed by edges in categories $\{\Gamma_A : A \in \mathcal{A}', i \in A\}$;

8 p_i is the concatenation of (s_i, u_{e_i}), cycles starting/ending at u_{e_i} that traverse each of the above edge sequences, $(u_{e_i}, c_{i,1})$, and the rest of p'_i;

9 $\mathcal{G} = \bigcup_{i\in[n]} p_i$, where vertices in $\mathcal{G}_c$ (excluding sources/destinations) are labeled by their corresponding VNFs, all other vertices are labeled by the dummy f_0, and each w_A ($A \in \mathcal{A}$) is split evenly among edges in category Γ_A;

This is a supplementary file of [1].

Extension of Clique Embedding

We will show that one can build a feasible solution by extending algorithm Clique Embedding (CE).

Algorithm: Let $p'_i := s_i \oplus \mathbf{c}_i \oplus t_i$ be the service chain for flow d_i ($\oplus$: concatenation), augmented with its source and destination. Let $e_i := (s_i, c_{i,1})$ be the first-hop edge on p'_i, and $E_0 := \{e_i\}_{i\in[n]}$ be the set of distinct first-hop edges. As presented in Algorithm 1, the idea is to use one subgraph to represent the service chains (line 1) and another subgraph to represent the positive-weight categories not represented in the first subgraph (lines 2–3). Then each path p_i is formed by concatenating edge sequences that need to be traversed by flow d_i in both subgraphs (lines 6–8), as illustrated in Fig. 1.

Clearly, the constructed topology contains at most one instance per type of VNF. Meanwhile, each p_i is a valid path that traverses the service chain for flow d_i, and each category with positive weight contains at least one edge. Note that the category of a removed edge $e \in E_0$ is represented by its replacement edges $(s(e), u_e)$ and $(u_e, t(e))$.

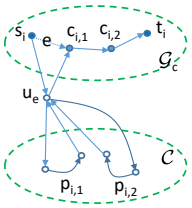


Fig. 1. Illustration of Clique Embedding++.

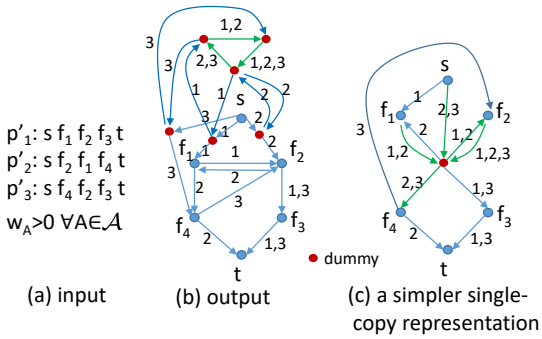


Fig. 2. Suboptimality of CE++ (edge label denotes its category index).

Remark: Given an $O(n^2)$-sparse solution to the weight inference problem (Proposition III.2), the above construction introduces $O(n)$ extra vertices (all labeled f_0) and $O(n^2)$ extra edges compared to the union of the service chains.

Suboptimality: While CE++ gives a feasible solution, it may introduce more vertices/edges than necessary. For example, consider the input in Fig. 2 (a), which specifies the service chains for three flows between the same source and the same destination. The output of CE++ is given in Fig. 2 (b), which has 12 vertices and 22 edges. However, Fig. 2 (c) is also a feasible solution, but has only 7 vertices and 11 edges. Our goal is to find the simplest solution in terms of the minimum order/size single-copy representation.

Note that our reason to introduce CE++ is to demonstrate that it is possible to augment a topology inferred from path length information alone to incorporate the information on service chains. As shown in the paper, a clean-slate solution (SAP) that jointly considers both types of information actually performs better in terms of inference accuracy.

Y. Lin (yjl5282@psu.edu) and T. He (tzh58@psu.edu) are with the Pennsylvania State University. S. Wang (wangshiq@us.ibm.com) is with IBM T. J. Watson Research Center. K. Chan (kevin.s.chan.civ@mail.mil) is with US Army Research Laboratory. S. Pasteris (s.pasteris@cs.ucl.ac.uk) is with University College London.

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.

References

[1] 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," *IEEE/ACM Transactions on Networking*, 2020, accepted for publication.