Overlay Routing Over an Uncooperative Underlay

Yudi Huang
Pennsylvania State University
State College, PA, USA
yxh5389@psu.edu

Ting He
Pennsylvania State University
State College, PA, USA
tinghe@psu.edu

ABSTRACT

Overlay network is a non-intrusive mechanism to enhance the existing network infrastructure by building a logical distributed system on top of a physical underlay. A major difficulty in operating overlay networks is the lack of cooperation from the underlay, which is usually under a different network administration. In particular, the lack of knowledge about the underlay topology and link capacities makes the design of efficient overlay routing extremely difficult. In contrast to existing solutions for overlay routing based on simplistic assumptions such as known underlay topology or disjoint routing paths through the underlay, we aim at systematically optimizing overlay routing without causing congestion, by extracting necessary information about the underlay from measurements taken at overlay nodes. To this end, we (i) identify the minimum information for congestion-free overlay routing, and (ii) develop polynomial-complexity algorithms to infer this information with guaranteed accuracy. Our evaluations in NS3 based on real network topologies demonstrate notable performance advantage of the proposed solution over existing solutions.

CCS CONCEPTS

• **Networks** → **Overlay and other logical network structures**; *Network performance modeling*; *Network measurement*.

KEYWORDS

overlay, routing, network tomography, capacity estimation

ACM Reference Format:
Yudi Huang and Ting He. 2023. Overlay Routing Over an Uncooperative Underlay. In *The Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc '23), October 23–26, 2023, Washington, DC, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3565287.3610274

1 INTRODUCTION

Overlay networks, referring to logical distributed systems running on top of a physical communication underlay, have been widely adopted to enhance the existing network infrastructure due to the difficulty of deploying infrastructure-wide upgrades. Frequently, overlay networks are used to provide value-adding functionalities

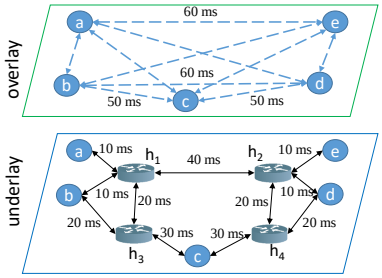


Figure 1: Example of underlay-aware overlay routing.

that a best-effort IP-based underlay network cannot provide, such as caching, traffic engineering, fast failover, and attack mitigation [21]. Meanwhile, the performance of an overlay network heavily relies on the proper control of overlay routing. For instance, caching overlay requires efficient routing between origin servers and edge servers to provide notable performance gain in the case of cache misses [21]. Large-scale applications spreading across multiple datacenters need careful routing of inter-datacenter flows to avoid congestion [4]. For mission-critical overlay applications, a proper selection of backup routes between overlay nodes that are maximally disjoint with primary routes is necessary for maintaining high Quality of Service (QoS) in the case of failures [8].

Due to its importance, tremendous efforts have been devoted to the design of overlay routing, e.g., [4, 8, 21, 24]. Compared to classical routing problems, one of the unique challenges in overlay routing is the lack of knowledge about the underlay, which can lead to incorrect overlay routing decisions. As a concrete example, consider the overlay-underlay network in Fig. 1, where link labels denote their (propagation) delays, and each overlay link maps to the shortest path (in delay) between its endpoints in the underlay. Suppose that the overlay needs to route two large flows with source-destination pairs (a, e) and (b, d), respectively. Further suppose that each link in the underlay has sufficient capacity for one of the flows but not both. Given the objective of minimizing the total delay, an underlay-agnostic routing algorithm that is only aware of the individual delays and capacities of overlay links will route both flows over the direct overlay paths $a \to e$ and $b \to d$. However, as these paths share a common link (h_1, h_2) in the underlay, this routing solution will cause congestion and hence poor performance. Meanwhile, an underlay-aware routing algorithm that has knowledge of how the overlay links share links in the underlay will choose the overlay paths $a \to e$ and $b \to c \to d$, which will minimize the total delay while avoiding congestion.

The need for overlay routing to be aware of the internal parameters of the underlay (e.g., topology, routing protocol, link characteristics) has been widely recognized. However, most of the existing works either assume such information to be directly provided by

MobiHoc '23, October 23–26, 2023, Washington, DC, USA

ACM ISBN 978-1-4503-9926-5/23/10…$15.00
https://doi.org/10.1145/3565287.3610274

the underlay [25, 27], or avoid explicitly requiring such knowledge by performing overlay routing on a trial-and-error basis [8]. The former approach is often inapplicable in practice due to the lack of cooperation from the underlay, and the latter approach is inefficient due to the exponentially large search space.

In this work, we aim at addressing these limitations by developing a framework for *underlay-aware overlay routing* that can systematically optimize the routing among overlay nodes without cooperation from the underlay. The core of our framework is a set of network inference algorithms that can extract the necessary information about the underlay from measurements within the overlay to enable overlay route optimization without congestion.

1.1 Related Work

Overlay routing: Overlay routing aims at controlling data forwarding among overlay nodes to optimize certain performance metrics while avoiding congestion. Typical performance metrics include routing cost [24], route update cost [25], and completion time of data transfer [4, 27]. Most of these works either assumed a cooperative underlay network whose internal parameters can be directly observed by the overlay [25, 27], or ignored the sharing of underlay links by the logical links between overlay nodes [4]. *In contrast, we address the more challenging problem of overlay routing over an uncooperative underlay network, while accounting for the underlay link sharing between overlay links.*

The work most related to ours is [28], which encoded the knowledge about the underlay as *linear capacity constraints (LCCs)*, which ensure that the total traffic load from the overlay does not exceed the capacity of any underlay link. However, [28] resorted to a simple heuristic based on [12] to infer the LCCs, which could only discover a small subset of LCCs that are insufficient for overlay routing. *In this regard, our contribution is a set of algorithms that can infer the minimum set of LCCs sufficient for overlay routing from observations at the overlay with guaranteed accuracy.*

Network tomography: One key piece of information for routing is network topology. In the face of an uncooperative underlay, the overlay has to infer its topology from measurements between overlay nodes, known as *network topology inference/tomography* [7]. However, topology inference is a challenging task by itself. Most existing solutions are based on the simplifying assumption that the routing paths for each source/destination form a tree; see [15] for a detailed review. The assumption of tree-based routing is frequently violated due to round-trip probing, load balancing, and network function traversals, but removing this assumption significantly complicates topology inference, for which only a few results exist [14, 15, 22]. Without the assumption of tree-based routing, the routing topology can no longer be uniquely identified from end-to-end measurements [15]. However, it is still possible to detect the existence of links shared by a subset of paths [15, 22], which turns out to be very useful for overlay routing as explained in Section 3.1. However, the existing solutions in [15, 22] both had exponential complexity. *In this regard, our contribution is the first polynomial-complexity algorithm for inferring how a set of arbitrary paths share links in a blackbox network from end-to-end measurements.*

Available capacity estimation: Another key piece of information for routing is the available link capacities (in the presence of background traffic). Available capacity estimation is a classical problem for which many tools have been developed; see [3]. In an uncooperative underlay, only tools based on end-to-end measurements are applicable. In this regard, existing works focused on inferring the available capacity at the bottleneck link of a given path, based on either the *probe gap model (PGM)* [16] or the *probe rate model (PRM)* [10]. To support overlay routing, *we leverage the existing single-path capacity estimation methods as subroutines and develop an algorithm to estimate the total available capacity over multiple paths with possibly shared links.* Our work is weakly related to *shared bottleneck detection (SBD)* [12], which aims at detecting which subset of flows share a bottleneck. However, most SBD methods only detect the existence of a shared bottleneck without estimating its available capacity. More importantly, SBD works under a given flow assignment and cannot characterize the feasible region for all the flow assignments, which is the focus of our work.

1.2 Summary of Contributions

We study the problem of overlay routing over an uncooperative underlay, with the following contributions:

1) We identify the minimum information about the underlay that is both sufficient for congestion-free overlay routing and uniquely identifiable from measurements between overlay nodes.

2) We develop the first polynomial-complexity algorithm to detect the existence of underlay links shared exclusively by each subset of paths between overlay nodes from end-to-end measurements, under arbitrary routing in the underlay. We also develop a greedy algorithm to estimate the effective capacity of the detected links based on single-path available capacity estimation methods.

3) We prove that our detection results have error probabilities that decay exponentially with the sample size, and our estimation results are no more than a constant factor away from the ground truth.

4) We test our solution against benchmarks via packet-level simulations in NS3 based on real network topologies and link parameters. Our results show that despite facing inference errors, our algorithms can still better characterize the feasible region for overlay routing than existing solutions, which leads to notably less congestion and better communication performance.

Roadmap. Section 2 formulates our problem, for which Section 3 addresses the inference about the underlay, and Section 4 addresses the overlay routing. Both solutions are evaluated in Section 5. Finally, Section 6 concludes the paper. **All the proofs can be found in the Supplementary Material.**

2 PROBLEM FORMULATION

2.1 Network Model

The underlay network is modeled as a connected undirected graph $\underline{G} = (\underline{V}, \underline{E})$, where $\underline{V}$ denotes the set of underlay nodes and $\underline{E}$ the set of underlay links. Each link $\underline{e} \in \underline{E}$ has a finite capacity $C_{\underline{e}}$.

The overlay network, managed by a centralized entity such as an *overlay network operation center (ONOC)* [5] or a software defined wide area network (SD-WAN) controller [24], is modeled as a connected directed graph $G = (V, E)$, where $V \subseteq \underline{V}$ is the set of nodes that are part of the overlay (e.g., running the overlay application), and each overlay link $e = (i, j) \in E$ denotes a tunnel between two overlay nodes that maps to the underlay routing path $\underline{p}_{i,j}$ from

node i to node j. We do not impose any limiting assumption on the underlay routes, and allow asymmetric routing (i.e., $\underline{p}_{i,j}$ and $\underline{p}_{j,i}$ may not be the same). In the sequel, we will use "tunnel" and "overlay link" interchangeably.

Remark: The assumption of centralized management of the overlay is used to study the overlay routing problem without worrying about coordination within the overlay; the extension to distributed solutions is left to future work.

2.2 Objective of Overlay Routing

Given a set of flow demands H, the goal of overlay routing is to optimally satisfy these demands by controlling the data forwarding among overlay nodes. We consider an uncooperative underlay by assuming that: (i) the overlay can control how to route its flows among the overlay nodes, but not how to route between adjacent overlay nodes within the underlay; (ii) the overlay can observe the overlay topology G and the parameters of overlay links, but not the underlay topology $\underline{G}$, its routing paths $\{\underline{p}_{i,j}\}_{(i,j)\in E}$, or the parameters of underlay links.

In the above context, a basic need of the overlay is to route its flows to optimize certain performance metric of interest, subject to capacity constraints imposed by the underlay. As a concrete example, consider the objective of minimizing the overlay routing cost as formulated below. Suppose that each flow demand $h \in H$ specifies a source $s_h \in V$, a destination $t_h \in V$, and a fixed flow rate d_h. Sending a unit of flow over tunnel $(i, j) \in E$ incurs a routing cost of $c_{ij} \geq 0$, which can model considerations like bandwidth leasing cost or QoS degradation cost (e.g., delay). The overlay can control how the flows traverse overlay nodes through a decision variable $x_{ij}^h \in \{0, 1\}$, which indicates whether flow $h \in H$ traverses tunnel (i, j) (in the direction of $i \to j$). Define b_i^h as 1 if $i = s_h$, -1 if $i = t_h$, and 0 otherwise. The *minimum cost overlay routing* problem can be formulated as follows:

$$\min_{\boldsymbol{x}} \quad \sum_{(i,j)\in E} c_{ij} \sum_{h\in H} d_h x_{ij}^h \tag{1a}$$

$$\text{s.t.} \quad \sum_{(i,j)\in E:\, \underline{e}\in\underline{p}_{i,j}} \sum_{h\in H} d_h x_{ij}^h \leq C_{\underline{e}}, \quad \forall \underline{e} \in \underline{E}, \tag{1b}$$

$$\sum_{j\in V} x_{ij}^h = \sum_{j\in V} x_{ji}^h + b_i^h, \quad \forall h \in H, i \in V, \tag{1c}$$

$$x_{ij}^h \in \{0, 1\}, \quad \forall h \in H, (i, j) \in E. \tag{1d}$$

The objective (1a) is the total routing cost for the overlay. Constraint (1b) is the *per-link capacity constraint* to ensure that the load on each underlay link is within its capacity, constraint (1c) is the flow conservation constraint to ensure that the overlay links in $\{(i, j) \in E : x_{ij}^h = 1\}$ form a path from s_h to t_h ($\forall h \in H$), and constraint (1d) ensures that only one path is selected for each flow (assuming single-path routing is required). Therefore, the optimal solution to (1) provides the set of overlay paths to route the flows in H that achieves the minimum routing cost without causing congestion.

The optimization (1) is NP-hard, as it is a generalization of the *minimum-cost multiple-source unsplittable flow problem (MMUFP)* that is NP-hard [1]. Nevertheless, as an integer linear programming (ILP) problem, it can be tackled by a number of heuristics developed for MMUFP, e.g., greedy and LP relaxation with randomized rounding [1], and the optimal solution can also be computed for small instances by existing ILP solvers via algorithms such as branch-and-price-and-cut [2].

Remark 1: The formulation (1) is just an example of the possible objectives of overlay routing. Other formulations can also be considered. For instance, in addition to the routing cost (1a), there may also be a cost in setting up tunnels as considered in [24], and instead of fixing the flow rate d_h, the overlay may want to design d_h to finish data transfer as soon as possible [4, 27]. We will focus on the formulation (1) in this work for concreteness and leave the study of other formulations to future work.

Remark 2: The optimization (1) assumes that there exists at least one solution $\boldsymbol{x}$ that can satisfy all the demands in H within the capacity constraint (1b), i.e., (1) is feasible. When this assumption is violated, we can relax the constraint (1b) into

$$\sum_{(i,j)\in E:\, \underline{e}\in\underline{p}_{i,j}} \sum_{h\in H} d_h x_{ij}^h \leq C_{\underline{e}}\omega, \quad \forall \underline{e} \in \underline{E}, \tag{2}$$

by introducing a new variable $\omega \geq 1$ to denote the maximum overloading factor for the underlay links. We can ensure feasibility while discouraging overloading by adding a penalty term "$c_\omega(\omega - 1)$" to the objective function (1a), where the parameter $c_\omega \geq 0$ controls the tradeoff between cost and congestion. Setting c_ω to a large value will make congestion avoidance the primary objective and cost minimization the secondary objective, which reduces the relaxed formulation to (1) in underloaded cases and to a *minimum overload overlay routing* problem, i.e., min ω s.t. (2), (1c), (1d), in overloaded cases.

2.3 Problem Statement

From (1), we can see that overlay routing depends on the underlay primarily through the capacity constraint (1b), which requires two pieces of information: (i) how the tunnels are routed through the underlay $(\underline{p}_{i,j})_{(i,j)\in E}$, and (ii) the underlay link capacities $(C_{\underline{e}})_{\underline{e}\in\underline{E}}$. While the overlay may have other considerations requiring further information about the underlay, satisfying the capacity constraint is a basic requirement, and is thus the focus of our work.

Compared to routing in flat networks, the main challenge for routing in overlay networks is the lack of information about the underlay. In contrast to existing works on overlay routing that resorted to either the underlay's cooperation or heuristic inference methods to obtain the information they required (see Section 1.1), we aim at developing a complete solution that infers the minimum information needed for overlay routing based on measurements at overlay nodes with guaranteed accuracy, and then optimizes overlay routing based on the inferred information, using the minimum cost overlay routing problem (1) as a concrete example.

3 OVERLAY-BASED INFERENCE

We will first analyze the minimum information the overlay needs about the underlay and then address how to infer this information.

3.1 Minimum Information for Overlay Routing

A straightforward implementation of (1) requires detailed knowledge of the underlay topology in terms of the routes $(\underline{p}_{i,j})_{(i,j)\in E}$

and the link capacities $(C_{\underline{e}})_{\underline{e}\in\underline{E}}$, in order to formulate constraint (1b). A natural question is thus whether we can directly apply solutions from topology inference to obtain this information. At a first look, the answer seems negative without further assumptions, because topology inference faces an inherent ambiguity that the routing topology capable of generating a given set of end-to-end measurements is generally not unique [15]. However, to support overlay routing, there is actually no need to infer the underlay topology. Instead, it suffices to infer just enough information to compute the feasible region defined by constraint (1b).

To formalize this idea, we introduce the following notion, adapted from [15, 22] to our problem.

Definition 3.1. *A **category of links traversed by** F **out of** E ($F \subseteq E$) is the set of underlay links traversed by and only by the tunnels in F out of all the tunnels in E, i.e,*[1]

$$\Gamma_F(E) \coloneqq \Big(\bigcap_{(i,j)\in F} \underline{p}_{i,j}\Big) \setminus \Big(\bigcup_{(i,j)\in E\setminus F} \underline{p}_{i,j}\Big). \tag{3}$$

A straightforward implication of the above definition is that the paths measurable by the overlay induce the following partition of the underlay links:

$$\underline{E} = \bigcup_{F\subseteq E} \Gamma_F(E). \tag{4}$$

For example, in Fig. 1, if E contains all the tunnels between the nodes $\{a, b, c, d, e\}$, then link $(h_1, h_2) \in \Gamma_F(E)$ for $F \coloneqq \{(a, e), (e, a), (a, d), (d, a), (b, e), (e, b), (b, d), (d, b)\}$, because (h_1, h_2) is traversed by all the tunnels in F but no other tunnel in E.

Our key observation is that since all the links in the same category are traversed by the same set of tunnels, they must carry the same traffic load from the overlay. Therefore, we can reduce the per-link capacity constraint (1b) to the following *per-category capacity constraint*:

$$\sum_{(i,j)\in F}\sum_{h\in H} d_h x_{ij}^h \le C_F, \quad \forall F \subseteq E \text{ with } \Gamma_F(E) \neq \emptyset, \tag{5}$$

where C_F, referred to as the *category capacity*, is the minimum capacity of all the links in category $\Gamma_F(E)$, i.e.,

$$C_F \coloneqq \min_{\underline{e}\in\Gamma_F(E)} C_{\underline{e}}. \tag{6}$$

The new constraint (5) is equivalent to the original constraint (1b) in that an overlay routing solution satisfies one of these constraints if and only if it satisfies the other. However, instead of requiring detailed information about the underlay (i.e., $(\underline{p}_{i,j})_{(i,j)\in E}$ and $(C_{\underline{e}})_{\underline{e}\in\underline{E}}$), *implementing constraint* (5) *only requires the knowledge of the nonempty categories and their capacities.*

3.2 Detection of Nonempty Categories

The detection of nonempty categories from end-to-end measurements has been used as an intermediate step in topology inference [15, 22]. The idea is to define an additive metric such that the path-level metrics can be estimated from end-to-end measurements and the category-level metrics can be estimated from the path-level metrics. Then under the following assumption, we can detect the nonempty categories as those with non-zero metrics.

[1]We abuse the notation a little to use $\underline{p}$ to denote the set of links traversed by path $\underline{p}$.

Assumption 1. *All nonempty categories have non-zero metrics.*

This assumption holds as long as all the underlay links have positive metrics, which intuitively means that every link imposes non-zero performance degradation (e.g., loss, delay, delay variation) to packets traversing it. This assumption is reasonable, as a link with no impact on communication performance will not be detectable from end-to-end measurements.

3.2.1 Defining Additive Metrics. We first need to define a performance metric θ. such that: (i) the link metrics are nonnegative and additive, and (ii) the corresponding path metrics can be reliably inferred from end-to-end performance measurements. Following [15], we adopt a metric of the form:

$$\theta_{\underline{e}} \coloneqq -\log \alpha_{\underline{e}}, \tag{7}$$

where $\alpha_{\underline{e}} \in (0, 1)$ denotes the probability for a packet transmitted over link $\underline{e}$ to experience the "good state", with different versions of this metric for different definitions of $\alpha_{\underline{e}}$. For example, if $\alpha_{\underline{e}}$ is the probability for a packet to successfully traverse $\underline{e}$ without being lost, then (7) is the *loss-based metric* [18], and if $\alpha_{\underline{e}}$ is the probability for a packet to traverse $\underline{e}$ without incurring queueing delay, then (7) is the *utilization-based metric* [18].

To make this metric additive, we assume that the states of different underlay links are independent of each other, which is a common assumption in topology inference [15, 18, 19, 22]. Moreover, to discover shared links, we adopt a commonly-used probing method of sending batches of concurrent probes over all the tunnels. Due to the fact that packets arriving at a link in quick succession experience very similar performance, probes in the same batch are assumed to experience the same link state when traversing a shared link, which is again a common assumption [15, 17, 18, 22]. Let $S_F \in \{0, 1\}$ indicate whether the probes in a batch experience good states on all the tunnels in $F \subseteq E$. As $S_F = 1$ if and only if all the underlay links in $\bigcup_{(i,j)\in F} \underline{p}_{i,j}$ are in good states, we have

$$\begin{aligned}\rho_F \coloneqq -\log \Pr\{S_F = 1\} &= -\log\left(\prod_{\underline{e}\in\bigcup_{(i,j)\in F}\underline{p}_{i,j}} \alpha_{\underline{e}}\right)\\ &= \sum_{\underline{e}\in\bigcup_{(i,j)\in F}\underline{p}_{i,j}} \theta_{\underline{e}},\end{aligned} \tag{8}$$

which means that $\theta_{\underline{e}}$ defined in (7) is an additive metric over a union of simultaneously probed paths. Here ρ_F denotes the metric for the union of paths for the tunnels in F, which can be estimated consistently by the overlay from observations of S_F.

Remark: The assumptions of independent states at different links and identical states at a shared link for probes in the same batch are simplifying assumptions that may not hold strictly in practice. Nevertheless, solutions derived from these assumptions have been validated in Internet experiments [18]. We will stress-test our solution derived from these assumptions in NS3 simulations where the assumptions may not hold (see Section 5).

3.2.2 Inferring Category Metrics. The overlay cannot directly generate equation (8) as it does not know the routing path $\underline{p}_{i,j}$ for

each tunnel $(i, j) \in E$. Nevertheless, the overlay can utilize the estimate of ρ_F to infer the following information about the categories without any knowledge of the routing paths.

Definition 3.2. *For a given category $\Gamma_F(E)$, the associated **category metric** $w_F(E)$ is defined as the sum metric for all the links in category $\Gamma_F(E)$, i.e., $w_F(E) \coloneqq \sum_{\underline{e}\in\Gamma_F(E)} \theta_{\underline{e}}$.*

The key is to note that by the definition of category, we have

$$\bigcup_{(i,j)\in F} \underline{p}_{i,j} = \bigcup_{F'\subseteq E: F'\cap F\neq\emptyset} \Gamma_{F'}(E), \quad \forall F \subseteq E, \tag{9}$$

which allows (8) to be rewritten as an equation of category metrics:

$$\rho_F = \sum_{F'\subseteq E:\ F'\cap F\neq\emptyset} w_{F'}(E), \quad \forall F \subseteq E. \tag{10}$$

Equations like (10) can be generated without prior knowledge of the underlay topology. Moreover, these equations are known to uniquely determine the category metrics.

Theorem 3.1 (Theorem III.1 in [14]). *Given the path metrics $(\rho_F)_{F\subseteq E, F\neq\emptyset}$, the category metrics $(w_F)_{F\subseteq E,F\neq\emptyset}$ are uniquely determined by* (10).

This theorem, together with the fact that link metrics affect path metrics only through category metrics, implies that the category metrics are the metrics of the finest granularity that can be uniquely identified by the overlay.

Example: Consider the network in Fig. 1. If only considering the tunnels in $E = \{(a, e), (a, d)\}$, we can partition the traversed underlay links into three nonempty categories: $\Gamma_{F_1}(E) = \{(h_2, e)\}$ for $F_1 \coloneqq \{(a, e)\}$, $\Gamma_{F_2}(E) = \{(h_2, d)\}$ for $F_2 \coloneqq \{(a, d)\}$, and $\Gamma_E(E) = \{(a, h_1), (h_1, h_2)\}$ (the other links are in category $\Gamma_\emptyset(E)$). Thus, the category metrics are $w_{F_1}(E) = \theta_{(h_2,e)}$, $w_{F_2}(E) = \theta_{(h_2,d)}$, and $w_E(E) = \theta_{(a,h_1)} + \theta_{(h_1,h_2)}$. Based on (10), we have a linear system:

$$\rho_{F_1} = w_E(E) + w_{F_1}(E), \tag{11a}$$

$$\rho_{F_2} = w_E(E) + w_{F_2}(E), \tag{11b}$$

$$\rho_E = w_E(E) + w_{F_1}(E) + w_{F_2}(E), \tag{11c}$$

which uniquely determines $(w_{F_1}(E), w_{F_2}(E), w_E(E))$. Meanwhile, the same path metrics $(\rho_{F_1}, \rho_{F_2}, \rho_E)$ can be generated by many different topologies (e.g., there may be multiple links between h_2 and e, or the tunnels (a, e) and (a, d) may join/branch multiple times) as long as the category metrics $(w_{F_1}(E), w_{F_2}(E), w_E(E))$ remain the same, making the category metrics the finest granularity information that the overlay can reliably infer from its measurements.

3.2.3 Taming Exponential Complexity. A straightforward solution for detecting nonempty categories based on solving (10) faces a severe limitation that the complexity grows at $O(2^{|E|})$, where $|E| = O(|V|^2)$, as the number of equations/variables is $O(2^{|E|})$. This renders the straightforward solution inapplicable beyond overlays with just a few nodes. To address this limitation, we develop a novel polynomial-complexity algorithm for category metric inference.

Our solution is based on dynamic programming. Instead of considering all the tunnels in one shot, we start with only a small subset of tunnels, for which (10) can be solved within acceptable time/space to obtain coarse-grained category metrics, and then we gradually expand the set of considered tunnels to refine the category metrics until all the tunnels are included. Our approach is motivated by the following observations:

Algorithm 1: Category Metric Inference

input : Set of all tunnels E, estimator of path metric ρ.
output: Non-zero category metrics $\{w_F(E) :\ w_F(E) \neq 0\}$

```
solve (10) to compute w(E_0) for an initial set of tunnels E_0 ⊆ E;
for t = 1, ..., |E| − |E_0| do
    E_t ← E_{t−1} ∪ {e} for an arbitrary tunnel e ∈ E \ E_{t−1};
    w_{e}(E_t) ← ρ_{E_t} − ρ_{E_{t−1}};
    for F ∈ supp(w(E_{t−1})) in increasing order of |F| do
        w_{F∪{e}}(E_t) ← ρ_{(E_{t−1}\F)∪{e}} − ρ_{E_{t−1}\F} − w_{e}(E_t) −
          Σ_{F'⊂F:F'∈supp(w(E_{t−1}))} w_{F'∪{e}}(E_t);
        w_F(E_t) ← w_F(E_{t−1}) − w_{F∪{e}}(E_t);
return {w_F(E_{|E|−|E_0|}) :  w_F(E_{|E|−|E_0|}) ≠ 0};
```

Lemma 3.1. *The number of nonempty categories is upper-bounded by the number of links in the underlay, i.e., $|\{\Gamma_F(E') : F \subseteq E', \Gamma_F(E') \neq \emptyset\}| \leq |\{\underline{e} : \underline{e} \in \cup_{(u,v)\in E'} \underline{p}_{u,v}\}| \leq |\underline{E}|$ for any $E' \subseteq E$.*

Lemma 3.2. *For any $E' \subset E$ and $e \in E \setminus E'$, $w_F(E') = 0$ implies $w_F(E' \cup \{e\}) = w_{F\cup\{e\}}(E' \cup \{e\}) = 0$, for all $F \subseteq E'$ and $F \neq \emptyset$.*

Lemma 3.3. *For any $E' \subset E$ and $e \in E \setminus E'$, $w_F(E') = w_F(E' \cup \{e\}) + w_{F\cup\{e\}}(E' \cup \{e\})$.*

Lemma 3.1 means that the vector of category metrics is sparse, and Lemma 3.2 means that the sparsity pattern of this vector for a subset of tunnels can be used to estimate its sparsity pattern as we consider more tunnels. Lemma 3.3 allows us to use the previously computed category metrics defined for a subset of tunnels to solve for the new category metrics when considering one more tunnel.

Algorithm: Based on the above observations, we develop a dynamic programming algorithm for computing the non-zero category metrics for any given set of tunnels, as shown in Algorithm 1. We ignore estimation error in ρ. for now to focus on the main idea; how to handle estimation error will be discussed later. Here, E_t denotes the set of tunnels considered in iteration t, $\boldsymbol{w}(E_t) \coloneqq (w_F(E_t))_{F\subseteq E_t, F\neq\emptyset}$, and $\mathrm{supp}(\boldsymbol{w}(E_t)) \coloneqq \{F \subseteq E_t :\ F \neq \emptyset,\ w_F(E_t) \neq 0\}$. The algorithm first uses measurements from a small set of tunnels E_0 to compute a vector of coarse-grained category metrics $\boldsymbol{w}(E_0)$ by directly solving (10). It then gradually refines the solution by expanding the set of considered tunnels. In iteration t, the equations corresponding to $E_t = E_{t-1} \cup \{e\}$ can be classified into two types:

$$\rho_F = \sum_{F'\subseteq E_{t-1}, F'\cap F\neq\emptyset} \big(w_{F'}(E_t) + w_{F'\cup\{e\}}(E_t)\big), \tag{12}$$

$$\rho_{F\cup\{e\}} = \sum_{F'\subseteq E_{t-1}, F'\cap F\neq\emptyset} \big(w_{F'}(E_t) + w_{F'\cup\{e\}}(E_t)\big) + \sum_{F'\subseteq E_{t-1}\setminus F} w_{F'\cup\{e\}}(E_t), \tag{13}$$

where (12) is $\forall F \subseteq E_{t-1},\ F \neq \emptyset$ and (13) is $\forall F \subseteq E_{t-1}$. Given the solution $\boldsymbol{w}(E_{t-1})$ from the previous iteration, equations of type (12) become redundant, as their information is already contained in the simpler equations $w_F(E_{t-1}) = w_F(E_t) + w_{F\cup\{e\}}(E_t)$ based on

Lemma 3.3. Equations of type (13) can be rewritten as

$$\sum_{F' \subseteq F} w_{F' \cup \{e\}}(E_t) = \rho_{(E_{t-1} \setminus F) \cup \{e\}} - \rho_{E_{t-1} \setminus F}. \tag{14}$$

For $F = \emptyset$, (14) contains only one unknown variable $w_{\{e\}}(E_t)$, and hence can be used to compute $w_{\{e\}}(E_t)$ as in line 4. Based on this initial solution, we can use (14) to gradually solve $w_{F \cup \{e\}}(E_t)$ in the increasing order of $|F|$ as in line 6, because when we try to solve $w_{F \cup \{e\}}(E_t)$, the values of $w_{F' \cup \{e\}}(E_t)$ for any $F' \subset F$ have been obtained. Once $w_{F \cup \{e\}}(E_t)$ is obtained, we can apply Lemma 3.3 to compute $w_F(E_t)$ as in line 7. In this process, we use the observation in Lemma 3.2 to reduce complexity by only computing $w_F(E_t)$ and $w_{F \cup \{e\}}(E_t)$ for $F \subseteq E_{t-1}$ satisfying $F \neq \emptyset$ and $w_F(E_{t-1}) \neq 0$. Note that $E_{|E|-|E_0|} = E$.

Complexity: Algorithm 1 significantly improves the complexity of category metric inference compared to the straightforward solution. Specifically, under perfect estimation of the path metrics, each iteration (lines 2–7) incurs $O(|\underline{E}|^2)$ operations, stores $O(|\underline{E}|)$ variables, and performs $O(|\underline{E}|)$ estimations of path metrics, because the number of non-zero category metrics $|\text{supp}(\boldsymbol{w}(E_{t-1}))| \leq |\underline{E}|$ by Lemma 3.1. As there are $O(|E|)$ iterations, the total complexity is $O(|E| \cdot |\underline{E}|^2)$ in time, $O(|\underline{E}|)$ in space (reused across iterations), and $O(|E| \cdot |\underline{E}|)$ in the number of path metric estimations.

Handling errors: In practice, errors in the estimated path metrics $\hat{\rho}$. may cause the inferred category metrics $\hat{\boldsymbol{w}}(E_t)$ to be non-zero for more than $|\underline{E}|$ categories. If an upper bound $|\underline{\hat{E}}|$ on the number of underlay links is known, we can enforce $|\text{supp}(\hat{\boldsymbol{w}}(E_t))| \leq |\underline{\hat{E}}|$ at the end of each iteration by setting all but the top $|\underline{\hat{E}}|$ values to zero. Alternatively, we can perform a hypothesis test for each inferred category metric $\hat{w}_F(E_t)$ to determine whether $w_F(E_t) = 0$ (and set $\hat{w}_F(E_t)$ to zero if so), for which several existing tests can be applied [22].

Further speedups: Another idea for reducing the complexity of category metric inference is to filer out empty categories based on link sharing information, originally proposed in [22]. The basic idea is that since $\Gamma_F(E) \neq \emptyset$ only if there is at least one link shared by all the tunnels in F, we can set $w_F(E) = 0$ if $\exists F' \subseteq F$ that does not have any shared link, i.e., $\bigcap_{(i,j) \in F'} \underline{p}_{i,j} = \emptyset$. Specifically, pairwise link sharing between two paths can be detected easily, e.g., by testing whether their delays are correlated. We can then filter out empty categories by setting $w_F(E) = 0$ if $\exists (i,j), (i',j') \in F$ such that $\underline{p}_{i,j} \cap \underline{p}_{i',j'} = \emptyset$ (indicated by having zero delay covariance). While applying such filtering alone will not reduce the complexity sufficiently (the remaining number of categories can still be large), we can combine the filtering with our dynamic programming algorithm to achieve further speedup. Generally, given a collection of tunnel sets $\Psi \coloneqq \{F \subseteq E : \bigcap_{(i,j) \in F} \underline{p}_{i,j} = \emptyset\}$ known to have no common link, we can incorporate this information into Algorithm 1 by adding $w_{F'}(E_t) \leftarrow 0$ after line 6 (where $F' = F \cup \{e\}$) and line 7 (where $F' = F$) if $\exists F'' \subseteq F'$ such that $F'' \in \Psi$. In the special case of $|F| = 2$ for all $F \in \Psi$, adding this filtering step increases the time complexity to $O(|E| \cdot |\underline{E}| \cdot (|\underline{E}| + |E|^2))$ in theory, but in practice can actually accelerate Algorithm 1 by reducing the number of variables while improving the accuracy.

3.2.4 Performance Analysis. We now quantify the error in detecting nonempty categories using the inferred category metrics. Let $\eta > 0$ denote the detection threshold such that category $\Gamma_F(E)$ is detected as nonempty if and only if its inferred metric $\hat{w}_F(E) > \eta$. To gain explicit insights, our analysis will focus on the vanilla case where $\hat{\boldsymbol{w}}(E)$ is obtained by directly solving (10) based on the estimated path metrics $\hat{\boldsymbol{\rho}}$. The modifications introduced in Section 3.2.3 make it difficult to obtain explicit insights through analysis, and thus will be evaluated empirically (see Section 5).

All the errors originate from the error in estimating the path metric ρ_F defined in (8). As common in the literature [17, 18], we assume that ρ_F is estimated by plugging the empirical probability $\overline{S}_F \coloneqq \frac{1}{T} \sum_{t=1}^{T} S_{F,t}$ into (8):

$$\hat{\rho}_F \coloneqq -\log \overline{S}_F, \tag{15}$$

where $S_{F,t} \in \{0, 1\}$ indicates whether the probes in the t-th batch experience good states on all the tunnels in F. We now analyze the error in nonempty category detection as a function of the sample size T and other parameters.

We start by deriving the solution to (10) in closed form.

Lemma 3.4. *Each category metric is related to the path metrics by*

$$w_F(E) = \sum_{F' \subseteq F} (-1)^{|F'|+1} \rho_{(E \setminus F) \cup F'}, \ \forall F \subseteq E, F \neq \emptyset. \tag{16}$$

We then analyze the error in estimating ρ_F by (15). Let $s_F \coloneqq \Pr\{S_F = 1\}$ for ease of presentation.

Lemma 3.5. *For $T \gg 1$, the bias of* (15) *satisfies*

$$\mathbb{E}[\hat{\rho}_F] - \rho_F \approx \frac{1 - s_F}{2 s_F T}, \tag{17}$$

and the variance satisfies

$$\text{var}[\hat{\rho}_F] \approx \frac{1 - s_F}{s_F T}, \tag{18}$$

where smaller terms at the order of $o(1/T)$ have been ignored.

By the central limit theorem, the distribution of $\overline{S}_F$ is asymptotically Gaussian. While due to the nonlinear transform $-\log(\cdot)$, the distribution of $\hat{\rho}_F$ is not exactly Gaussian, the delta method [26] suggested that it is well approximated by the Gaussian distribution for large T. Formally, the delta method [26] states that for a sequence of random variables $(X_n)_{n \geq 1}$ satisfying $\sqrt{n}(X_n - \mu) \xrightarrow{D} \mathcal{N}(0, \sigma^2)$ and a function $f(x)$ such that the first derivative $f'(x)$ exists and is non-zero, we have

$$\sqrt{n}(f(X_n) - f(\mu)) \xrightarrow{D} \mathcal{N}\left(0, (f'(\mu))^2 \sigma^2\right), \tag{19}$$

where $\xrightarrow{D}$ denotes the convergence in distribution. Our problem satisfies these conditions with $\sqrt{T}(\overline{S}_F - s_F) \xrightarrow{D} \mathcal{N}(0, s_F(1 - s_F))$, $f(x) = -\log(x)$, $f(\overline{S}_F) = \hat{\rho}_F$, and $f(s_F) = \rho_F$. Under the Gaussian approximation, we can analyze the error in nonempty category detection in closed form as follows.

Theorem 3.2. *Suppose that the path metric estimation errors* $\{\hat{\rho}_F - \rho_F\}_{F\subseteq E, F\neq\emptyset}$ *can be modeled as independent Gaussian random variables with mean and variance given by Lemma 3.5. If* $w_F(E) = 0$*, then*

$$\Pr\{\hat{w}_F(E) > \eta\} = 1 - \Phi\left(\frac{\eta\sqrt{T} - \tilde{\delta}_F(E)/\sqrt{T}}{\delta_F(E)}\right) \tag{20}$$

$$\approx \frac{\delta_F(E)}{\eta\sqrt{2\pi T}} \exp\left(-\frac{\eta^2}{2\delta_F(E)^2}T\right), \tag{21}$$

and if $w_F(E) > \eta$*, then*

$$\Pr\{\hat{w}_F(E) \leq \eta\} = \Phi\left(\frac{(\eta - w_F(E))\sqrt{T} - \tilde{\delta}_F(E)/\sqrt{T}}{\delta_F(E)}\right) \tag{22}$$

$$\approx \frac{\delta_F(E)}{(w_F(E) - \eta)\sqrt{2\pi T}} \exp\left(-\frac{(\eta - w_F(E))^2}{2\delta_F(E)^2}T\right), \tag{23}$$

where $\Phi(\cdot)$ *is the CDF of the standard Gaussian distribution,*

$$\delta_F(E) := \sqrt{\sum_{F':E\setminus F\subseteq F'} (1 - s_{F'})/s_{F'}}, \tag{24}$$

$$\tilde{\delta}_F(E) := \sum_{F':E\setminus F\subseteq F'} (-1)^{|F'|-|E\setminus F|+1}(1 - s_{F'})/(2s_{F'}), \tag{25}$$

and the "$\approx$*" in* (21) *and* (23) *holds for* $T \gg 1$*.*

Remark: Theorem 3.2 states that both the false alarm probability (21) and the miss probability (23) decay exponentially with the sample size T, with the error exponent controlled by the detection threshold η. The threshold η essentially controls what kinds of categories are detectable, in the sense that a category must have at least one link in the "bad state" (e.g., with backlogged queue) with probability $> 1 - e^{-\eta}$ to be detectable with exponentially decaying error.

3.3 Estimation of Category Capacities

We now address the estimation of the capacity for each detected nonempty category. In the sequel, we will simply denote a category $\Gamma_F(E)$ as Γ_F and a category metric $w_F(E)$ as w_F since they are always defined with respect to all the tunnels in E.

In absence of any prior knowledge, the overlay has to measure the category capacities. However, the minimum link capacity C_F for a nonempty category Γ_F will not be measurable by the overlay if no flow assignment in the overlay can saturate the minimum-capacity link. To address this issue, we define a notion called *effective category capacity* $\tilde{C}_F$ as follows.

Definition 3.3. *For each* $F \subseteq E$*, the* effective category capacity $\tilde{C}_F$ *is the maximum flow that can be sent through the tunnels in* F*, i.e.,*

$$\tilde{C}_F := \max_{(f_e)_{e\in E}} \sum_{e\in F} f_e \tag{26a}$$

$$s.t. \sum_{e'\in F'} f_{e'} \leq C_{F'}, \quad \forall F' \subseteq E,\ \Gamma_{F'} \neq \emptyset, \tag{26b}$$

$$f_e \geq 0, \quad \forall e \in E. \tag{26c}$$

The effective category capacity is equivalent to the category capacity defined in (6) in that they induce the same feasible region for overlay routing, except that the effective category capacity is always achievable by the overlay. Thus, it suffices for the overlay to estimate the effective capacity of each nonempty category.

Algorithm 2: Effective Category Capacity Estimation

input : set $\mathcal{F}$ of category indices of interest (e.g., $\mathcal{F} := \{F \subseteq E : \hat{w}_F > \eta\}$)
output : Estimated effective category capacities $\{\hat{C}_F\}_{F\in\mathcal{F}}$

1 **for** *each* $F := \{e_{i_1}, \cdots, e_{i_{|F|}}\} \in \mathcal{F}$ **do**
2 $f_{e_{i_1}} \leftarrow \hat{C}_{e_{i_1}}(\mathbf{0})$;
3 **for** $j = 2, \cdots, |F|$ **do**
4 $f_{e_{i_j}} \leftarrow \hat{C}_{e_{i_j}}(\boldsymbol{f})$;
5 $\hat{C}_F \leftarrow \sum_{j=1}^{|F|} f_{e_{i_j}}$;
6 return $\{\hat{C}_F\}_{F\in\mathcal{F}}$;

Although how to estimate the combined capacity over multiple tunnels (i.e., paths) has not been solved systematically, how to estimate the available capacity of a single tunnel has been well understood [10, 16]. Thus, we will build on top of these existing solutions to develop an algorithm for estimating $\tilde{C}_F$. Our algorithm assumes *a subroutine that can estimate the residual capacity of a tunnel* e *under an existing flow assignment* $\boldsymbol{f}$, which can be implemented by any of the existing available capacity estimation methods. Let $\tilde{C}_e(\boldsymbol{f})$ denote the true residual capacity of e under $\boldsymbol{f}$ and $\hat{C}_e(\boldsymbol{f})$ the estimate given by the subroutine.

Algorithm: Given this subroutine, we propose an algorithm in Algorithm 2. For each tunnel set of interest F, this algorithm goes through the tunnels in F in an arbitrary order, and tries to assign as much flow as possible onto each tunnel e_{i_j} according to the residual capacity estimated by the subroutine, without backtracking the flow assignment for $e_{i_1}, \ldots, e_{i_{j-1}}$ (lines 2–4). The effective category capacity is then estimated as the sum flow (line 5).

Complexity: As Algorithm 2 invokes an existing single-path available capacity estimation method as subroutine, its exact complexity will depend on the complexity of the subroutine. Nevertheless, as the complexity of the subroutine is independent of the size of the overlay-underlay network, we can analyze the complexity of Algorithm 2 in terms of the number of invocations of the subroutine, which equals $O(|\mathcal{F}| \cdot |E|)$. As the number of nonempty categories is upper-bounded by the number of underlay links $|\underline{E}|$, under reasonably accurate nonempty category detection, the number of detected nonempty categories $|\mathcal{F}|$ will be $O(|\underline{E}|)$, and thus the complexity of Algorithm 2 will be $O(|\underline{E}| \cdot |E|)$.

Accuracy: We now analyze how the estimated effective category capacity $\hat{C}_F$ provided by Algorithm 2 compares to the true value. Under the assumption that the subroutine does not overestimate the residual capacities of individual tunnels, which is typical for PGM-based methods [16], it is easy to see that the flow assignment in Algorithm 2 is feasible for the underlay link capacities, i.e., feasible for (26). Thus, the achieved sum rate can only underestimate the effective category capacity, i.e., $\hat{C}_F \leq \tilde{C}_F$. Meanwhile, if the subroutine is accurate, then the estimate can only be a constant factor smaller as stated below.

Theorem 3.3. *If the estimation for single-tunnel residual capacity is accurate (i.e.,* $\hat{C}_e(\boldsymbol{f}) = \tilde{C}_e(\boldsymbol{f})$*), then Algorithm 2 achieves* $1/q_F$*-approximation. More precisely,* $\tilde{C}_F \geq \hat{C}_F \geq \tilde{C}_F/q_F$*, where*

$$q_F := \max_{e\in F} |\{F' \subseteq E : e \in F', \Gamma_{F'} \neq \emptyset, |F' \cap F| > 1\}| \tag{27}$$

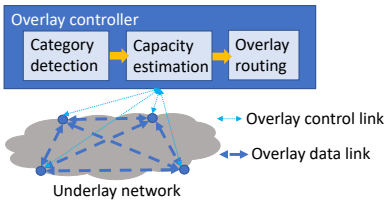


Figure 2: Illustration of overall solution.

is the maximum number of nonempty categories a tunnel in F traverses that are shared by at least another tunnel in F.

Even if the subroutine incurs error, Algorithm 2 still achieves a constant-factor approximation under the following condition.

Corollary 3.3.1. *If the estimate $\hat{C}_e(f)$ for any single-tunnel residual capacity $\tilde{C}_e(f)$ satisfies $\tilde{C}_e(f) \geq \hat{C}_e(f) \geq \tilde{C}_e(f)/q$, then Algorithm 2 achieves $1/(q \cdot q_F)$-approximation. More precisely, $\tilde{C}_F \geq \hat{C}_F \geq \tilde{C}_F/(q \cdot q_F)$, where q_F is defined in* (27).

Remark: We have shown in [9] that the approximation ratio in Theorem 3.3 is tight, i.e., there exist instances where Algorithm 2 underestimates the effective capacity of a category by a factor arbitrarily close to $1/q_F$. However, we have observed that the worst case rarely occurs, and Algorithm 2 is usually accurate as long as its subroutine for estimating single-tunnel residual capacity is accurate.

4 UNDERLAY-AWARE OVERLAY ROUTING

4.1 Overall Solution

By detecting the nonempty categories $\mathcal{F}$ (via Algorithm 1) and estimating the effective category capacities $\{\hat{C}_F\}_{F\in\mathcal{F}}$ (via Algorithm 2), the overlay can generate capacity constraints in the form of

$$\sum_{(i,j)\in F} \sum_{h\in H} d_h x_{ij}^h \leq \hat{C}_F, \quad \forall F \in \mathcal{F}, \tag{28}$$

and use them in place of (1b) in overlay routing optimizations such as (1). Fig. 2 illustrates the workflow of the overall proposed solution. Note that the centralized controller is just an illustration of the fact that the current work does not focus on the coordination within the overlay (which is left to future work).

4.2 Performance Analysis

Both nonempty category detection and category capacity estimation are subject to inference errors, which will affect the accuracy of the generated constraint (28) and thus the performance of overlay routing. We now analyze the impact of these errors.

There are four types of inference errors: false alarm/miss in nonempty category detection and under/over-estimation of category capacity. A false alarm in nonempty category detection will cause the generation of a superfluous constraint in overlay routing, which may lead to suboptimal routing decisions. Meanwhile, a miss will cause a constraint to be missing, which may lead to an infeasible routing decision that causes congestion in the underlay. Similarly, an underestimated category capacity will lead to a constraint that is too tight, potentially causing suboptimality, while an overestimated category capacity will lead to a constraint that is too loose, potentially causing congestion. While the extent of suboptimality will depend on the specific routing objective and network instance, which is hard to characterize analytically, the congestion probability can be analyzed in closed form. Specifically, as discussed in Section 3.3, the category capacity estimation typically incurs only underestimation errors, which will not cause congestion. Thus, the only cause of congestion is the failure in detecting some nonempty category, the probability of which will decay exponentially in T (#batches of probes for estimating the path metrics) as follows.

	AttMpls	AboveNet	GTS-CE	BellCanada
$\lvert\underline{V}\rvert$	25	23	149	48
$\lvert\underline{E}\rvert$	114	62	386	130
$C_{\underline{e}}$ (Gbps)	1	1	1	1
link delays (μs)	[206,4973]	[100, 13800]	[5,1081]	[78, 6160]

Table 1: Characteristics of the tested underlay topologies.

Theorem 4.1. *Let $\mathcal{F}^* := \{F \subseteq E : \Gamma_F \neq \emptyset\}$ be the true set of nonempty categories. Suppose that the (effective) category capacity estimation is performed by Algorithm 2 with a subroutine for single-tunnel residual capacity estimation that has no overestimation error, and every nonempty category satisfies $w_F > \eta$, where η is the threshold for nonempty category detection. Then under the assumption of Theorem 3.2, the probability for the proposed underlay-aware overlay routing to cause congestion is upper-bounded by*

$$|\mathcal{F}^*|\Phi\left(\frac{(\eta - w_{F^*})\sqrt{T} - \tilde{\delta}_{F^*}/\sqrt{T}}{\delta_{F^*}}\right) \approx \frac{|\mathcal{F}^*|\delta_{F^*}}{(w_{F^*} - \eta)\sqrt{2\pi T}} \exp\left(-\frac{(w_{F^*} - \eta)^2}{2\delta_{F^*}^2} T\right), \tag{29}$$

where δ_F, $\tilde{\delta}_F$, and Φ are defined as in Theorem 3.2 (omitting "(E)"), and $F^* := \arg\max_{F\in\mathcal{F}^*} \Pr\{\hat{w}_F \leq \eta\}$.

5 PERFORMANCE EVALUATION

5.1 Evaluation Setup

In this section, we will test the proposed solutions via packet-level simulations in NS3, which is a widely used discrete event simulator. To construct diverse and realistic scenarios, we simulate the underlay network according to four real networks from the Internet Topology Zoo [13] with different densities and sizes, and set the link capacities and delays according to [6]. The characteristics of each topology are summarized in Table 1.

Each underlay is assumed to follow shortest path routing based on hop count. Following [11], we generate cross traffic on each underlay link according to an ON-OFF process, where the duration of each ON period follows a truncated Pareto distribution, with shape parameter 2.04 and scale/upper-bound parameter set to the minimum/maximum round-trip time (RTT) of the tunnels traversing this link. The duration of each OFF period follows the same distribution with a different scale parameter, configured to yield a link utilization randomly drawn from [10%, 40%]. Following [23], we randomly draw the sizes of cross-traffic packets from 50, 576, and 1460 bytes with probabilities 0.4, 0.2, and 0.4, respectively. We set the overlay packet size to 50 bytes for probing and 1000 bytes for routing.

To create the overlay, we select 10 nodes with the lowest degree as the overlay nodes while maintaining a pairwise distance of at least two hops, which leads to 90 (directed) overlay tunnels and 2^{90} potential categories. The number of nonempty categories for each

	AttMpls	AboveNet	GTS-CE	BellCanada
#empty cat.	$2^{90}-69$	$2^{90}-51$	$2^{90}-59$	$2^{90}-51$
#nonempty cat.	69	51	59	51
#false alarms	603	542	2159	1695
#misses	20	25	40	27

Table 2: Misses and false alarms in category detection.

topology is given in Table 2. As for the demands $\boldsymbol{d} = (d_h)_{h\in H}$ in (1), we first generate an initial demand $\boldsymbol{d}_0$ based on the gravity model [20]. Then, we scale it by a factor α to ensure that there exists a routing solution to satisfy $\alpha\boldsymbol{d}_0$ with a given maximum link utilization.

5.2 Benchmarks

We evaluate the following solutions:

(1) "Agnostic": a baseline that treats all the tunnels as independent logical links, i.e., ignoring their sharing of links;
(2) "LCC": the state-of-the-art solution from [28], under two optimistic assumptions: (i) perfect clustering of the detected flows based on their shared dominant bottlenecks, and (ii) improved accuracy in the capacity constraints based on the residual capacities instead of the total capacities as in [28];
(3) "Proposed": our proposed solution as depicted in Section 4.1;
(4) "Enhanced proposed": an enhanced version of our solution with two added sets of constraints: one set from "LCC" for fair comparison (due to the optimistic assumptions given to it) and another set obtained by running Algorithms 1–2 for each set of tunnels sharing the same source.

5.3 Evaluation Results

5.3.1 Nonempty Category Detection. We estimate ρ_F as in (15), where a probe is considered to experience good state on a tunnel if its delay is below a threshold. To determine the threshold, we profile the delay on each tunnel during light traffic and set the threshold as the mean delay plus three standard deviations. We select 9 tunnels with the same (randomly selected) source to form E_0. A category Γ_F is detected to be nonempty if its inferred metric satisfies $\hat{w}_F > \eta$. As our estimation of the effective category capacities is accurate (see Table 3), false alarms will not hurt overlay routing, and thus we set η to a small value (10^{-5} in our simulation) to minimize misses. The resulting numbers of false alarms/misses are given in Table 2, which are the median of 20 Monte Carlo runs, each containing 2×10^4 batches of probes. Despite the large number of false alarms, the false alarm rate is very low due to the exponentially many categories that are empty. Meanwhile, we observe a high miss rate, primarily due to the error in estimating ρ_F. Such errors come from two phenomena: (i) for tunnels with different sources, probes in the same batch may arrive at a shared link at different times and experience different queueing delays; (ii) a link shared by a large number of tunnels will receive many probes in a batch, where the earlier probes will experience different queueing delays from the later ones.

5.3.2 Category Capacity Estimation. Next, we evaluate the normalized mean absolute error ($|\hat{C}_F - \tilde{C}_F|/\tilde{C}_F$) of Algorithm 2. To separate the impact of errors in its subroutine, we evaluate two versions of Algorithm 2, one using the true value of $C_{e_{ij}}(\boldsymbol{f})$ in Line 5 (i.e., ideal subroutine) and the other using the estimated $\hat{C}_{e_{ij}}(\boldsymbol{f})$ obtained from *pathload*[2] [10]. The results averaged over 20 Monte Carlo runs are given in Table 3. Surprisingly, Algorithm 2 can estimate the effective category capacities almost perfectly when the subroutine estimates the single-tunnel residual capacities correctly, indicating that the worst-case ratio in Theorem 3.3 is rarely achieved. Slightly more error is incurred when using a realistic subroutine, but the overall estimation remains highly accurate.

	AttMpls	AboveNet	GTS-CE	BellCanada
ideal subroutine	0.10%	0.13%	0.13%	0.4%
pathload	1.07%	1.18%	1.15%	1.49%

Table 3: Errors in effective category capacity estimation.

5.3.3 Approximation of Feasible Region. Despite the large numbers of misses and false alarms, the feasible region induced by the inferred capacity constraint (28) may still approximate the true feasible region induced by (1b) (equivalently (5)), if the superfluous constraints caused by false alarms have similar effect as the missing constraints caused by misses. Denote the true feasible region of the rates through the tunnels as $\mathcal{P} := \{\boldsymbol{y} \geq \boldsymbol{0} : \sum_{(i,j)\in E:\underline{e}\in\underline{p}_{i,j}} y_{ij} \leq C_{\underline{e}},\ \forall \underline{e} \in \underline{E}\}$, and the inferred feasible region as $\hat{\mathcal{P}} := \{\boldsymbol{y} \geq \boldsymbol{0} : \sum_{(i,j)\in F} y_{ij} \leq \hat{C}_F,\ \forall F \in \mathcal{F}\}$. We define $\hat{\mathcal{P}}$ similarly for each of the benchmarks. We measure the consistency between these two regions by randomly sampling extreme points from one region and calculating the maximum constraint violation for the other region. We observe that the extreme points of $\mathcal{P}$ almost always satisfy the constraints of $\hat{\mathcal{P}}$ for all the solutions (omitted), but the extreme points of $\hat{\mathcal{P}}$ can violate the constraints of $\mathcal{P}$ (i.e., causing congestion), as shown in Fig. 3. We see from Fig. 3 that (i) the constraint violation of "Agnostic" is most severe, (ii) our proposed solution notably reduces the constraint violation compared to both "Agnostic" and "LCC", and (iii) the enhancement to our solution can further reduce the constraint violation but only slightly. *In summary, despite the notable inference errors, our solution can still characterize the feasible region for overlay routing much more accurately than the existing solutions.*

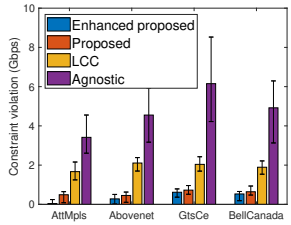
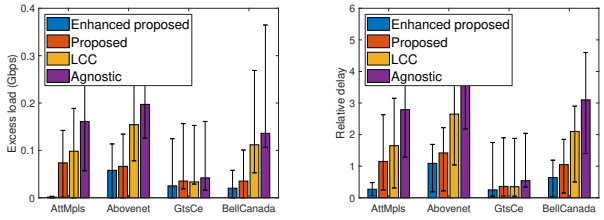


Figure 3: Constraint violation for randomly-sampled extreme points of the estimated feasible region.

5.3.4 Performance of Overlay Routing. When the demands are sufficiently light such that even "Agnostic" does not encounter any congestion, there is no need to consider link sharing among tunnels and all the overlay routing solutions achieve similar performance (omitted). We thus focus on scenarios where at least one link will be congested under one of the tested routing solutions, by scaling the

[2]Pathload is an adaptive algorithm that sends a train of probes at a time and tunes its rate to measure the residual capacity. In our simulation, the train length is set to 5000 probes, but the total number of trains is a variable in [2, 15].



(a) max excess load **(b) avg relative delay**

Figure 4: Performance of overlay routing.

demands to achieve a maximum link utilization of 90% under perfect knowledge about the underlay. We evaluate the performance of minimum cost overlay routing in terms of both congestion and routing cost. Here, we set the routing cost c_{ij} for each tunnel $(i, j) \in E$ to the sum (propagation) delay of the links traversed by this tunnel.

To measure congestion, we evaluate the maximum load on any underlay link in excess of its capacity. The result in Fig. 4 (a) shows that "Agnostic" incurs the most congestion due to ignoring link sharing among the tunnels, followed by "LCC" that only considers a subset of the capacity constraints corresponding to the bottlenecks shared by tunnels with the same destination. Our proposed solution and its enhanced version can notably reduce the congestion, thanks to their better accuracy in approximating the feasible region (Fig. 3). These observations also apply to the sum of excess loads.

To measure routing cost, we simulate overlay routing for 20,000 milliseconds and measure the average end-to-end delay over all the received packets, repeated for 20 Monte Carlo runs. We then normalize the average delay: given the average delay $\bar{\phi}$ obtained from simulation, we evaluate $(\bar{\phi} - \bar{\phi}_0)/\bar{\phi}_0$, where $\bar{\phi}_0$ is the average delay under the optimal routing solution based on perfect knowledge about the underlay. The result in Fig. 4 (b) confirms that (i) underlay-aware overlay routing (our solutions and "LCC") can notably outperform underlay-agnostic overlay routing ("Agnostic"), and (ii) by inferring the key information to characterize the feasible region, our solutions can substantially outperform "LCC" for well-connected underlays (AttMpls, Abovenet, BellCanada). Meanwhile, all the solutions perform similarly for sparsely-connected underlays (GtsCe).

6 CONCLUSION

We studied the problem of overlay routing over an uncooperative underlay with unknown topology and link capacities. We identified the minimum information needed by the overlay for congestion-free overlay routing, and then developed polynomial-complexity algorithms to infer this information with guaranteed accuracy. Our NS3 simulations based on realistic settings demonstrated the superior performance of our algorithms in characterizing the feasible region and improving the performance of overlay routing.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under award CNS-2106294 and CNS-1946022.

REFERENCES

[1] Yasuhito Asano. 2000. Experimental Evaluation of Approximation Algorithms for the Minimum Cost Multiple-source Unsplittable Flow Problem.. In *ICALP Satellite Workshops*. 111–122.

[2] Cynthia Barnhart, Christopher A Hane, and Pamela H Vance. 2000. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research* 48, 2 (2000), 318–326.

[3] Shilpa Shashikant Chaudhari and Rajashekhar C Biradar. 2015. Survey of bandwidth estimation techniques in communication networks. *wireless personal communications* 83, 2 (2015), 1425–1476.

[4] Li Chen, Shuhao Liu, and Baochun Li. 2021. Optimizing Network Transfers for Data Analytic Jobs Across Geo-Distributed Datacenters. *IEEE Transactions on Parallel and Distributed Systems* 33, 2 (2021), 403–414.

[5] Yan Chen, David Bindel, Han Hee Song, and Randy H Katz. 2007. Algebra-based scalable overlay network monitoring: algorithms, evaluation, and applications. *IEEE/ACM Transactions on Networking* 15, 5 (2007), 1084–1097.

[6] Steven Gay, Pierre Schaus, and Stefano Vissicchio. 2017. Repetita: Repeatable experiments for performance evaluation of traffic-engineering algorithms. *arXiv preprint arXiv:1710.08665* (2017).

[7] Ting He, Liang Ma, Ananthram Swami, and Don Towsley. 2021. *Network Tomography: Identifiability, Measurement Design, and Network State Inference.* Cambridge University Press.

[8] Thomas Holterbach, Stefano Vissicchio, Alberto Dainotti, and Laurent Vanbever. 2017. Swift: Predictive fast reroute. In *SIGCOMM*. 460–473.

[9] Yudi Huang and Ting He. 2023. Overlay Routing Over an Uncooperative Underlay. https://sites.psu.edu/nsrg/files/2023/03/Yudi_Mobihoc23_TechReport.pdf

[10] Manish Jain and Constantinos Dovrolis. 2003. End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput. *IEEE/ACM Transactions on networking* 11, 4 (2003), 537–549.

[11] Hao Jiang and Constantinos Dovrolis. 2005. Why is the internet traffic bursty in short time scales?. In *SIGMETRICS*. 241–252.

[12] Dina Katabi and Charles Blake. 2001. Inferring congestion sharing and path characteristics from packet interarrival times. *MIT Report* (2001).

[13] Simon Knight, Hung X Nguyen, Nickolas Falkner, Rhys Bowden, and Matthew Roughan. 2011. The internet topology zoo. *IEEE Journal on Selected Areas in Communications* 29, 9 (2011), 1765–1775.

[14] Y. Lin, T. He, S. Wang, K. Chan, and S. Pasteris. 2019. Multicast-Based Weight Inference in General Network Topologies. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. 1–6. https://doi.org/10.1109/ICC.2019.8761099

[15] Yilei Lin, Ting He, Shiqiang Wang, Kevin Chan, and Stephen Pasteris. 2020. Looking glass of NFV: Inferring the structure and state of NFV network from external observations. *IEEE/ACM Transactions on Networking* 28, 4 (2020), 1477–1490.

[16] Xiliang Liu, Kaliappa Ravindran, and Dmitri Loguinov. 2008. A stochastic foundation of available bandwidth estimation: Multi-hop analysis. *IEEE/ACM Transactions on Networking* 16, 1 (2008), 130–143.

[17] Jian Ni and Sekhar Tatikonda. 2011. Network tomography based on additive metrics. *IEEE Transactions on Information Theory* 57, 12 (2011), 7798–7809.

[18] Jian Ni, Haiyong Xie, Sekhar Tatikonda, and Yang Richard Yang. 2009. Efficient and dynamic routing topology inference from end-to-end measurements. *IEEE/ACM transactions on networking* 18, 1 (2009), 123–135.

[19] Michael G Rabbat, Mark J Coates, and Robert D Nowak. 2006. Multiple-source Internet tomography. *IEEE Journal on Selected Areas in Communications* 24, 12 (2006), 2221–2234.

[20] Matthew Roughan. 2005. Simplifying the synthesis of Internet traffic matrices. *ACM SIGCOMM* 35, 5 (2005), 93–96.

[21] Ramesh K Sitaraman, Mangesh Kasbekar, Woody Lichtenstein, and Manish Jain. 2014. Overlay networks: An Akamai perspective. *Advanced Content Delivery, Streaming, and Cloud Services* 51, 4 (2014), 305–328.

[22] Kevin D Smith, Saber Jafarpour, Ananthram Swami, and Francesco Bullo. 2022. Topology Inference With Multivariate Cumulants: The Möbius Inference Algorithm. *IEEE/ACM Transactions on Networking* (2022).

[23] Ales Svigelj, Mihael Mohorcic, Gorazd Kandus, Ales Kos, Matevz Pustisek, and Janez Bester. 2004. Routing in ISL networks considering empirical IP traffic. *IEEE Journal on Selected areas in Communications* 22, 2 (2004), 261–272.

[24] Diman Zad Tootaghaj, Faraz Ahmed, Puneet Sharma, and Mihalis Yannakakis. 2020. Homa: An efficient topology and route management approach in SD-WAN overlays. In *IEEE INFOCOM*. 2351–2360.

[25] Hai-Anh Tran, Duc Tran, and Abdelhamid Mellouk. 2022. State-Dependent Multi-Constraint Topology Configuration for Software-Defined Service Overlay Networks. *IEEE/ACM Transactions on Networking* (2022).

[26] Jay M Ver Hoef. 2012. Who invented the delta method? *The American Statistician* 66, 2 (2012), 124–127.

[27] Yuchao Zhang, Junchen Jiang, Ke Xu, Xiaohui Nie, Martin J Reed, Haiyang Wang, Guang Yao, Miao Zhang, and Kai Chen. 2018. BDS: a centralized near-optimal overlay network for inter-datacenter data replication. In *EuroSys*. 1–14.

[28] Ying Zhu and Baochun Li. 2008. Overlay networks with linear capacity constraints. *IEEE Transactions on Parallel and Distributed systems* 19, 2 (2008), 159–173.