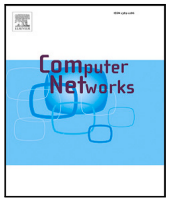
Computer Networks 200 (2021) 108489

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



A Holistic Approach for Locating Traffic Differentiation in the Internet

Thiago Garrett [a,*], Luis C. E. Bona [b], Elias P. Duarte Jr. [b]

[a] University of Oslo, Norway
[b] Federal University of Paraná, Curitiba, Brazil

ARTICLE INFO

Keywords:
Network Neutrality
Traffic Differentiation
Internet peering

ABSTRACT

The worldwide debate over Network Neutrality (NN) has been raging on for nearly two decades. According to NN principles, all traffic in the Internet must be treated with impartiality. In particular, unfair Traffic Differentiation (TD) is not allowed. Several strategies have been proposed for detecting TD, but locating the source of TD is still an under-explored topic. In this work, we present a holistic approach for unifying TD detection solutions into a single framework with the purpose of locating the source of TD. We propose an algorithm for combining measurements from multiple vantage points, and a strategy for selecting good vantage points. Our proposals leverage Internet peering properties to infer the behavior of individual Autonomous Systems (ASes), without requiring knowledge of the exact routes traversed by measurement probes. To evaluate our proposals, we first ran several experiments to confirm that indeed Internet routes do present the required properties. Then, several simulations were performed to assess the efficiency of our proposals. Results show that our approach is capable of locating TD under several different conditions. Another finding is that issuing measurements from a few end-hosts of core Internet ASes achieves similar results than from a much larger number of end-hosts at the edge.

1. Introduction

Network Neutrality (NN) has been the focus of hot debates around the world since Tim Wu coined the term in 2002 [1]. NN states that all traffic in the Internet must be treated with impartiality, regardless of its origin, destination and/or content. This effectively means that unfair Traffic Differentiation (TD), such as prioritizing or degrading specific traffic flows, are prohibited [2]. Arguments in favor of the implementation of this principle claim that TD may threaten the open nature of the Internet as an environment that fosters innovation, fair competition, and consumer's freedom of choice [3,4]. Arguments against NN claim that giving Internet Service Providers (ISPs) more freedom to manage their own networks fosters competition and innovation [5,6].

Several countries around the world have established NN regulations for preventing unfair TD [7]. However, ISP compliance cannot be ensured by the regulations alone. Furthermore, even on a non-regulated environment, it is important to ensure the transparency of traffic management practices adopted by ISPs. In this context, there are several proposals in the literature for monitoring NN violations on the Internet [8]. These proposals focus mostly on detecting the presence of TD between end-hosts, employing a myriad of different measurement techniques and statistical methods. However, those solutions only detect TD, they are not capable of locating where exactly in the network discrimination was introduced. Locating TD is important to enforce regulations and to empower consumers by revealing potentially discriminatory behaviors from certain ISPs. To the best of our knowledge, there has been very little work on locating TD [9–12], all of which rely on unrealistic assumptions, such as prior knowledge of the complete network topology (at the host level), and knowledge about the precise paths traversed by measurement traffic.

In this work, we address the problem of locating TD with more realistic assumptions than those of previous works. In our proposal we recognize the fact that it may not be possible to know the exact host-level path between end-hosts, there may even be multiple paths, and the path actually traversed can change over time [13]. We take a holistic approach that aims at unifying the several existing TD detection solutions. We propose an algorithm that locates TD by combining TD detection measurements issued from multiple vantage points. By taking advantage of Internet peering properties instead of assuming complete knowledge about the paths between end-hosts, the proposed algorithm is able to infer which Autonomous System (AS) was responsible for discriminating traffic. We also propose a strategy for selecting vantage points that will effectively contribute to locate TD. Using both the proposed algorithm and the proposed strategy as building blocks, we finally describe a complete solution for locating TD in the Internet. An earlier version of the proposals and some of the results presented in this work were published as a conference paper in [14].

* Corresponding author.
E-mail addresses: thiagoga@ifi.uio.no (T. Garrett), bona@inf.ufpr.br (L.C.E. Bona), elias@inf.ufpr.br (E.P. Duarte Jr.).

https://doi.org/10.1016/j.comnet.2021.108489
Received 12 May 2021; Received in revised form 29 July 2021; Accepted 15 September 2021
Available online 25 September 2021

The proposed algorithm combines measurements taken by any existing TD detection solution – thus it can be seen as a "meta" strategy for aggregating measurements from multiple sources. By taking advantage of inter-AS routing properties, we list the possible AS-level paths between the measurement points [15]. Inference is made by checking which ASes are present in all possible paths between each pair of measurement hosts. For instance, if a given AS is in all paths between two hosts, and no TD was detected between them (i.e. as indicated by some TD detection tool), then it is possible to reach the conclusion that AS is not employing TD, since the measurement traffic surely went through it and yet no TD was detected.

The strategy for selecting measurement points searches for pairs of measurement points in such a way that all possible paths between the selected points traverse certain given ASes. The idea is that measurements issued from the selected points, when combined using the proposed algorithm, will effectively contribute to infer the behavior of the given ASes.

Finally, we present the complete solution for locating TD. The solution identifies which AS is discriminating traffic between a given pair of end-hosts. The main idea is to filter out the ASes that are not employing TD until only one AS remains — the one responsible for discriminating traffic. This is done by selecting measurement points using the strategy proposed, and combining the measurements issued from them using the proposed algorithm.

We performed two different sets of experiments to evaluate our proposals. We first executed experiments on the PlanetLab global testbed [16] to check whether our assumptions regarding the properties of AS-level paths are valid in the wild. Next, several simulations for assessing the efficiency of the solution for locating TD under different scenarios were executed. Results from the first set show that the peering properties we assume are valid for the majority of paths observed. Furthermore, results also reveal that path discovery techniques, employed by related work, may not return reliable results. Then the second set of experiments show that the proposed solution is capable of locating TD under different scenarios. Moreover, similar results were observed when combining measurements obtained with a large amount of ASes in the edge of the Internet and those from a small number of ASes in the core. The simulations also show which metrics should be employed to find more effective measurement points.

The main contributions of this paper are thus:

- This work advances the state of the art regarding the under-explored problem of locating TD in the Internet
- We propose an algorithm for inferring the behavior of ASes by combining measurements from multiple vantage points without requiring complete knowledge of the exact path between end-hosts
- A strategy for selecting measurement points that can effectively help locate TD is proposed
- A complete solution for locating TD that uses the proposed algorithm for combining measurements and the strategy for selecting measurements points is described and evaluated through simulation
- Metrics for selecting good measurement points are proposed and evaluated through simulation
- We show through simulation results the relation between the assumed Internet peering properties and the efficiency of our solution under different scenarios
- We make an innovative use of Internet peering properties
- We report experiments that show the limitations of path discovery techniques in the Internet, which are employed by related work
- We report experiments that show to which extent Internet peering properties are valid in the wild

The rest of this work is organized as follows. Section 2 presents related work. Then, an overview of key background concepts follows in Section 3. Next, we present the system model in Section 4. Then, the proposed algorithm for combining measurements is described in Section 5. Section 6 describes the proposed strategy for selecting measurements points. The complete solution for locating TD in the Internet is finally presented in Section 7. In Section 8, we describe the PlanetLab experiments for validating the assumed routing properties. Simulations for evaluating the complete solution for locating TD are presented in Section 9. Finally, we draw conclusions in Section 10.

2. Related work

Detecting TD in the Internet is a research topic widely explored in the literature [17–24]. A comprehensive survey [8] describes multiple TD detection solutions that rely on different types of network measurements and statistical methods. Measurement probes may be issued from one or several end-hosts. Other strategies rely on passive monitoring. The type of traffic employed by the probes may also differ. In general, TD detection is performed by comparing the measurements obtained to identify whether any set of measurements was statistically different from other sets, which may characterize a discriminatory treatment of network traffic. The solutions proposed in the present work make use of any of the existing TD detection solutions, rather than presenting yet a new alternative.

Few strategies have been proposed to *locate* where in the network discrimination was introduced. Most solutions [9–11] rely on path discovery techniques, in particular the *traceroute* tool [25]. *Traceroute* is supposed to discover the exact path traversed by measurement traffic between end-hosts. The idea is to try to detect the presence of TD on different subsets of the paths between end-hosts, in order to identify which portion of the path (or specific point) was responsible for introducing the discrimination. The major shortcomings of using these techniques is that *traceroute* is not reliable, since it is not always able to obtain the complete path traversed. Even when the path is discovered with precision, there is no guarantee that the measurement traffic traversed the same path as the application traffic under investigation. The experiments presented in Section 8 confirm some of the limitations of traceroute-like techniques. Our proposals for locating TD presented in this work take into account all the possible paths traffic may traverse between end-hosts, avoiding the shortcomings described above.

In [12] the authors propose an algorithm based on network tomography to detect TD and also locate in which host or link the discrimination occurred. The idea is to combine end-to-end measurements between several pairs of end-hosts. A system of equations is built with measurements as sums of intermediate values, each corresponding to a link traversed. Inconsistencies on the resolution of this system may indicate the presence of TD. The strategy relies on two strong assumptions: the exact host-level topology of the network is known, and as are the exact paths traffic traverses between all pairs of end-hosts. These assumptions may represent a problem in practice, as the host-level topology of the Internet is dynamic and inferences can be misleading. Furthermore, the path between a pair of end-hosts may change at any moment due to different reasons, e.g. load balancing, traffic exchange policies, router faults. In comparison, the present work only assumes knowledge of the AS-level topology, which can be easily obtained [26] – using Border Gateway Protocol (BGP) routing tables, for example. Moreover, our proposals do not assume that traffic always traverse the same path.

In another related work [27], we proposed an architecture for collecting and combining TD-related measurements from a plethora of sources. These include for instance, receiving inferences from a TD detection service running on the Cloud, or collecting NN-related measurements from an Internet of Things gateway. The architecture follows a hybrid active/passive approach, in which measurements are passively collected and combined, but active measurements can be requested on demand in order to investigate suspicious cases detected by aggregating the passive measurements. We argue that the proposals we introduce in the present work are possible directions for implementing that architecture.

3. Background

This section presents an overview of AS-level Internet routing properties, and describes a specific solution as an example of an effective TD detection solution that can be employed in our proposals.

3.1. AS-level routing properties

ASes are independent networks, owned by different organizations, each with a different set of assigned IP prefixes. The Internet consists of the interconnection of these independent networks. Several ASes may be traversed by any given traffic flow in the Internet. An AS-level path is defined as the sequence of ASes traversed by a data packet from the source end-host to the destination end-host. The path that is traversed by each packet depends both on its final destination and on the traffic exchange agreements each AS has with its neighbors. As a packet arrives, the AS decides to which neighboring AS the packet will be forwarded to. This decision is done by consulting a BGP table, which contains the neighboring ASes that can reach the final destination. One of these ASes is chosen according to a set of policies adopted.

ASes connect to other ASes in order to gain access to parts of the Internet which are not reachable directly from the local AS itself or through customer ASes. Traffic exchange agreements between ASes are not publicly available. However, it is possible to abstract the relationship between ASes into four categories [26]: peer-to-peer ($p2p$), sibling-to-sibling ($s2s$), customer-to-provider ($c2p$), and provider-to-customer ($p2c$). A $p2p$ relationship means that the two ASes exchange traffic between them and their customers without payments. When the two ASes are owned by the same company, they may exchange traffic freely in a $s2s$ relationship. In the case of $c2p$, a customer AS purchases transit services from a provider AS. Similarly, in a $p2c$ relationship an AS provides transit services to a customer AS, i.e. access to other parts of the network.

A widely accepted model for characterizing Internet paths is the Gao–Rexford model [15]. According to this model, AS-level paths in the Internet follow the *valley-free* property. In an AS-level path, ASes are customers of other ASes that provide transit services, and a transit provider AS is paid by the customer AS. An AS providing transit services without being paid by anyone configures a *valley* in the path, hence the name of the property. In a valley-free path, for each AS providing transit services there is a customer AS neighboring it, i.e. paying for the service. Thus the valley-free property states that valid paths in the Internet comply with the following pattern: any number of $c2p$ links, followed by up to one $p2p$ link, followed by any number of $p2c$ links. There may be any number of $s2s$ links anywhere along a path.

Fig. 1 shows an example of a real AS-level topology with the corresponding relationships, inferred by CAIDA [28]. In the figure, the path *Copel* → *RNP* → *UFPR* is valley-free, since the transit provider (*RNP*) is being paid by its customer (*UFPR*). However, *Copel* → *Sercomtel* → *Level 3* is a valley path, since no one is paying the transit provider *Sercomtel*.

Between any pair of end-hosts in the Internet, there may be several valley-free paths. Furthermore, each packet may traverse a different path, even packets of the same flow. This depends on the traffic exchange agreements in place and the routing policies of each AS [29]. The actual path traversed may also change over time [13]. For instance, in the topology shown in Fig. 1, paths *Sercomtel* → *Copel* → *Level 3* and *Sercomtel* → *ALGAR* → *Level 3* are both valley-free. In this particular case, the *Sercomtel* AS may prefer to exchange traffic with *ALGAR* through the $p2p$ link, since it would be cheaper than using the $c2p$ link with *Copel*.

We present experiments for evaluating to what extent the valley-free property is valid in the Internet in Section 8.

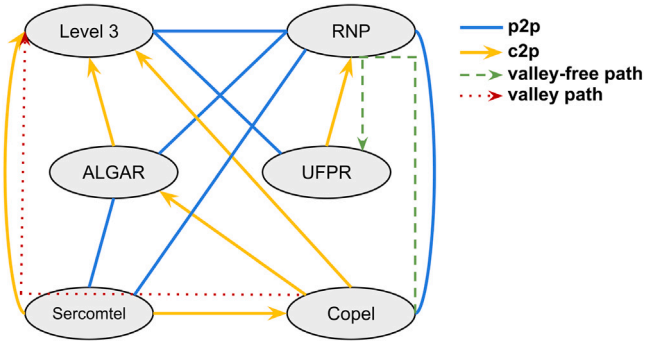


Fig. 1. Example of a real AS-level topology with the corresponding relationships, including valley and valley-free paths.

3.2. A representative solution for detecting TD

As described in Section 2, a large number of solutions for detecting TD have been proposed. In this work, TD is located by combining measurements issued by one or more of these solutions. As a confirmation that there are effective TD detection strategies, we give a brief overview of a representative solution next.

Wehe [23,24] is effectively able to detect TD by comparing measurements from two traffic flows transmitted between the same pair of end-hosts: a real traffic flow from an application under investigation, and that same traffic flow encrypted through a VPN tunnel. The authors report results based on a measurement campaign that lasted a whole year conducted by end-users with the solution running on their mobile devices. 1,045,413 measurements were obtained, from 126,249 users connected to 2735 different ISPs in 183 countries/regions. From the obtained data, the authors were able to perform a large-scale investigation of TD practices in the Internet. TD was detected in 7 different countries, and the majority of the TD cases detected affected video streaming services.

In the remainder of this work, we call *TD detectors* these solutions for detecting the presence of TD between end-hosts in the Internet. We argue that the inferences made by different TD detectors, and/or by multiple instances of the same TD detector, may complement each other to solve the problem of locating TD. Note that identifying whether the located discrimination is legal, beneficial, or not, is out of the scope of this work. We refer the reader to [30] for a detailed discussion about applications that require Quality of Service guarantees and NN regulations.

4. System model

In this section, we define the system model for locating TD. We first describe our assumptions in Section 4.1, followed by the model in Section 4.2. Finally, Section 4.3 describes how valley-free paths between ASes are obtained — which are a fundamental part of our proposals. Table 1 summarizes the main notations presented in this section and in the remainder of this work.

4.1. Assumptions

The proposed strategies for locating TD have requirements in terms of the AS-level Internet Topology assumed, routing properties of the network in which measurements are taken, the type of discrimination that can be located, and the availability of both TD detectors and measurement ASes. These assumptions are described next.

AS-level Internet Topology: The AS-level Internet topology is assumed to be known. Datasets that infer the Internet AS-level topology are publicly available, including the AS Rank project [28] by CAIDA, which is employed in this work. In particular, the relationships between ASes

Table 1
Table of notations.

Notation	Definition
I_u	Inferred behavior of AS u
$I_{u,v}$	Inferred behavior of the pair of ASes (u, v), as determined by a TD detector
M	Set of measurements $(u, v, I_{u,v})$ to be combined by the algorithm
D	Set of measurement ASes for a single execution of the solution
E	Initial AS pair for a single execution of the solution — ASes between which TD is to be located
σ	Maximum additional links relative to the shortest valley-free path considered when searching for paths
δ	Maximum valley-free distance from a suspect AS i up to the limit at which measurement ASes are to be checked
$V^{\sigma}_{u,v}$	Set of valley-free paths between ASes u and v with length bounded by σ
$A_{u,v}$	Set of all ASes present in the valley-free paths between u and v
Z	Set of initial pairs in a simulation scenario

was inferred from AS Rank data and employed to build an AS-level topology graph, as described in Section 8.

Valley-free Property: Another assumption is that the valley-free property is valid. This property is accepted as essential to guarantee the convergence of the BGP routing protocol [31]. In Section 8 we present experiments that assess to which extent this property is actually valid in the Internet.

Discrimination Types: In this work we assume TD based on packet content, such as application protocol, destination port, or even payload (through DPI — Deep Packet Inspection). TD based on the origin or destination of packets, for example, is out of the scope of this work. Recent work [24,32,33] reports results that indicate that TD based on content is common in the wild.

In this context, we assume that ASes always discriminate the same types of traffic, regardless of its origin or destination, or which ingress/egress points the traffic has traversed. As a consequence of this assumption, if traffic between a pair of end-hosts corresponding to a certain application is discriminated by an AS, traffic from the same application between any other pair of end-hosts will be discriminated by that AS as well.

TD Detectors: We assume the availability of at least one TD detector that is able to recognize the presence of TD between two given ASes. For instance, such TD detector can consist of an application running on end-hosts connected to the ASes, or a Virtualized Network Function (VNF) deployed within their networks. Note however that TD detectors are not able to locate TD (only detect). Detection alone leaves questions unanswered, as the paths between a given AS pair often consists of multiple ASes, any of which can be the responsible for TD. Locating TD is about pinpointing the exact one discriminating traffic.

Measurement ASes: We assume that a set of ASes are accessible for running TD detectors. We call them *measurement ASes*. In a real deployment, the set of available measurement ASes may change over time, for example if the set of connected end-hosts varies dynamically. However, in this work, we only focus on locating TD using the measurement ASes available during a specified period of time. Furthermore, we do not take into account specific characteristics of the end-hosts connected to measurement ASes (e.g. mobility, energy consumption).

4.2. System model

The AS-level topology of the Internet is represented as a directed graph $G = (A, L, f)$, where A is the set of ASes in the network and L is the set of connections between ASes. Let $R = \{c2p, p2c, p2p, s2s\}$ be the set of possible relationships between ASes: $f : L \rightarrow R$ is the function that maps a link $l \in L$ to the corresponding relationship $r \in R$.

The set of measurement ASes D is a subset of A, i.e. $D \subseteq A$. A path $p = \{u, \ldots, v\}$ is a sequence of ASes connecting ASes u and v. Let $P_{u,v}$ be the set of all paths between ASes u and v. Furthermore, L_p is the sequence of links of a path $p \in P_{u,v}$, and $R_p = \{f(l) \mid l \in L_p\}$ is the sequence of relationships between the corresponding ASes. A path $p \in P_{u,v}$ is valley-free if R_p follows the valley-free property as described in Section 3. The set of all valley-free paths between two ASes u and v is denoted by $V_{u,v} \subseteq P_{u,v}$.

We classify a given AS with respect to TD as either *discriminatory*, *neutral*, or *unknown*. An AS is classified as *discriminatory* if has been found to employ TD. Otherwise, it is classified as *neutral*. Finally, an AS is classified as *unknown* if after running all available solutions for detecting and locating TD it was not possible to infer whether it was employing TD or not. Let I_u be the inferred behavior of AS $u \in A$. Similarly, a pair of ASes (u, v), $u, v \in A$ is can be classified as either *neutral* or *discriminatory*. For a *neutral* pair of ASes, no end-to-end TD was detected between them, but if TD has been detected, then the pair is classified as *discriminatory*. Let $I_{u,v}$ be the inferred behavior of the pair of ASes (u, v), i.e. the output of a TD detector.

4.3. Searching for valley-free paths

A modified Depth-First Search (DFS) is employed in order to find the valley-free paths between ASes on the graph representing the topology. This search works as a traditional DFS, but discards all paths that do not follow the valley-free property. For this DFS, we employ a parameter (σ) that sets a maximum limit to the length of the paths. When searching for valley-free paths between two ASes, parameter σ corresponds to the number of additional links that can be added to the size of the shortest valley-free path between those ASes. The authors in [34] show that the complexity for listing all paths with bounded length between two vertices in a weighted directed graph is $O(mn + n^2 \log n)$, in which n is the number of vertices ($|A|$) and m is the number of edges ($|L|$).

For instance, with $\sigma = 1$, the DFS will find all shortest valley-free paths, as well as all valley-free paths one link larger than the shortest path. It is common for AS-level paths in the Internet to be larger than the shortest possible path. The experiments presented in Section 8 show to what extent this happens. This upper bound is important to keep the search computationally feasible. Let $p \in V_{u,v}$ be the shortest valley-free path between ASes u and v. We define $V^{\sigma}_{u,v} = \{p' \mid p' \in V_{u,v}, |p'| \leq |p| + \sigma\}$ as the set of valley-free paths between ASes u and v with length not larger than the length of the shortest path plus σ. Let $A_{e_1,e_2} = \bigcup \{p \mid p \in V^{\sigma}_{e_1,e_2}\}$ be the set of all ASes in all possible valley-free paths between e_1 and e_2.

Fig. 2 shows all the possible valley-free paths between the ASes e_1 and e_2, i.e. $V^{\sigma}_{e_1,e_2}$, for $\sigma = 0$. There are two possible paths between the pair: $\{e_1, c_1, c_3, e_2\}$, and $\{e_1, c_2, c_3, e_2\}$. Since we do not know which path would be effectively traversed by traffic between e_1 and e_2, we consider all possible paths. Moreover, in this example $A_{e_1,e_2} = \{e_1, e_2, c_1, c_2, c_3\}$.

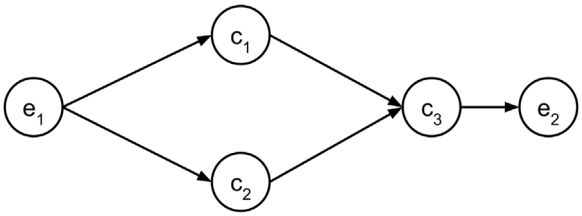


Fig. 2. Example of valley-free paths between ASes e_1 and e_2 ($V^{\sigma}_{e_1,e_2}$, $\sigma = 0$).

5. Algorithm for combining measurements

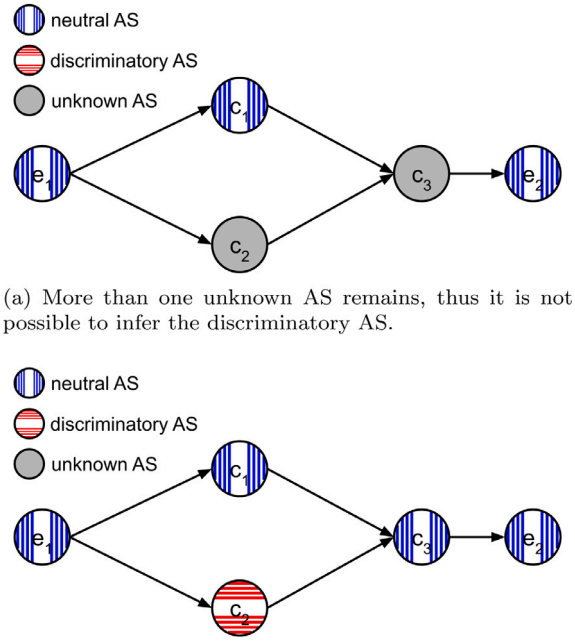
In this section, we present the algorithm proposed for combining inferences made by TD detectors. The goal of this algorithm is to identify which ASes are *neutral*, *discriminatory*, or *unknown*. The algorithm aggregates measurements from TD detectors running on multiple end-hosts – it is thus a holistic approach that relies on the outcome from any TD detector, including those not yet proposed. The main idea is to first identify neutral ASes, and then identify discriminatory ASes through a process of elimination.

The algorithm receives as input a set of measurements $M = \{(u_1, v_1, I_{u_1,v_1}), \ldots, (u_n, v_n, I_{u_n,v_n})\}$, in which each measurement (u_i, v_i, I_{u_i,v_i}) is a tuple containing two ASes (u_i and v_i) and the corresponding inferred behavior I_{u_i,v_i}. The algorithm outputs a tuple (N, C, U), in which N is the set of neutral ASes, C the set of discriminatory ASes, and U the set of unknown ASes. The algorithm consists of two steps. First, all the neutral AS pairs in M are evaluated to identify neutral ASes. The second step consists of a search for discriminatory ASes in M. A pseudo-code of the proposed algorithm is presented in Algorithm 1. We further describe the algorithm next.

Algorithm 1 Combining measurements.

1: $N \leftarrow \emptyset$; $C \leftarrow \emptyset$; $U \leftarrow \emptyset$ ▷ output sets
2: $M' \leftarrow \{(u, v, I_{u,v}) \mid (u, v, I_{u,v}) \in M, I_{u,v} = neutral\}$ ▷ neutral pairs
3: **for** each $(u, v, I_{u,v}) \in M'$ **do** ▷ first step
4: $A' \leftarrow \emptyset$
5: **for** each $p \in V^{\sigma}_{u,v}$ **do**
6: $A' \leftarrow A' \cup p$ ▷ All ASes in paths between u and v
7: **end for**
8: $T \leftarrow \{a \mid a \in A', \forall p \in V^{\sigma}_{u,v}, a \in p\}$ ▷ ASes present in all paths
9: $N \leftarrow N \cup T$ ▷ neutral ASes
10: $U \leftarrow U \cup A' \setminus T$ ▷ all other ASes are unknown
11: **end for**
12: $M'' \leftarrow \{(u, v, I_{u,v}) \mid (u, v, I_{u,v}) \in M, I_{u,v} = discriminatory\}$
13: **for** each $(u, v, I_{u,v}) \in M''$ **do** ▷ second step
14: $V' = \{p \setminus N \mid p \in V^{\sigma}_{u,v}\}$ ▷ paths after removing neutral ASes
15: $A'' \leftarrow \emptyset$
16: **for** each $p \in V'$ **do**
17: $A'' \leftarrow A'' \cup p$ ▷ all ASes in the remaining paths
18: **end for**
19: **if** $|A''| = 1$ **then**
20: $C \leftarrow C \cup A''$ ▷ only the discriminatory AS remains
21: **else**
22: $U \leftarrow U \cup A''$ ▷ more than one AS, so all are unknown
23: **end if**
24: **end for**
25: **return** (N, C, U)

The first step (lines 2–11) considers all measurements in M that indicated no TD. Let $M' = \{(u, v, I_{u,v}) \mid (u, v, I_{u,v}) \in M, I_{u,v} = neutral\}$ be the set of measurements which classified pairs of ASes as neutral. For each measurement $(u, v, I_{u,v}) \in M'$, a set T is created (line 8) containing all ASes that are present in all valley-free paths $p \in V^{\sigma}_{u,v}$. These ASes are classified as neutral, and added to the output set N (line 9). Note that T contains at least ASes u and v themselves. If any of these ASes had employed TD, $I_{u,v}$ would be *discriminatory* since the traffic from the TD detector would have surely traversed them. The rationale is that it is not possible to know which path the TD detector traffic actually took, thus the algorithm looks for the ASes that are present in *all* paths. For instance, in the example shown in Fig. 2, if TD was not detected for the pair (e_1, e_2) ($I_{e_1,e_2} = neutral$), then the ASes c_3, e_1, and e_2 would be inferred as neutral, since they are in all possible paths. However, nothing can be inferred for ASes c_1 and c_2, thus their classification remains unknown.



(a) More than one unknown AS remains, thus it is not possible to infer the discriminatory AS.

(b) Only one unknown AS remains, thus it is possible to infer the discriminatory AS.

Fig. 3. Example of measurements being combined to infer the behavior of ASes.

In the second step (lines 12–24), AS pairs for which TD was detected are evaluated. Let $M'' = \{(u, v, I_{u,v}) \mid (u, v, I_{u,v}) \in M, I_{u,v} = discriminatory\}$ be the set of measurements that classified AS pairs as discriminatory. For each measurement $(u, v, I_{u,v}) \in M''$, the algorithm removes from each possible path $p \in V^{\sigma}_{u,v}$ the ASes in N (line 14). Let $V' = \{p \setminus N \mid p \in V^{\sigma}_{u,v}\}$ be the set of valley-free paths that remain after removing the neutral ASes. If all non-empty paths $p' \in V'$ contain only a single AS c, then c is classified as discriminatory and added to C (lines 19–20). However, if there is more than one AS in the non-empty paths, they remain classified as unknown — it is not possible to know which of these ASes were traversed by traffic from the TD detector.

Fig. 3 shows two examples using the same pair of ASes as in Fig. 2. Suppose that TD was detected between e_1 and e_2. In Fig. 3(a), suppose that ASes e_1, e_2 and c_1 are already known to be neutral. Since there are two other ASes, c_2 and c_3, through which measurement traffic may have traversed, it is not possible to know which one of them is responsible for TD. Therefore, both c_2 and c_3 remain unknown. However, let us suppose then that c_3 is also known to be neutral. As shown in Fig. 3(b), the only remaining unknown AS would be c_2, thus it becomes possible to infer $I_{c_2} = discriminatory$.

The complexity of Algorithm 1 is derived as follows. The search for all valley-free paths is performed once for every measurement in M. Considering the complexity of listing all valley-free paths between two ASes with bounded length σ, as described in Section 4, the resulting complexity is $O(mno + n^2 o \log n)$, in which n is the number of vertices ($|A|$), m is the number of edges ($|L|$), and o is the number of measurements being combined by the algorithm ($|M|$).

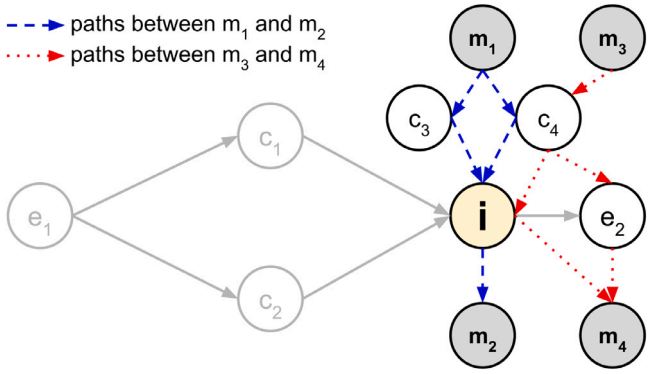


Fig. 4. Example of AS pair selection for investigating suspect AS i: pair (m_1, m_2) can be selected, while pair (m_3, m_4) cannot.

6. Strategy for selecting measurement points

In this section, we propose a strategy for selecting pairs of measurement ASes that can effectively help infer the behavior of a single given AS when TD measurements are combined. We call this single given AS a *suspect* AS. In order to be able to classify the behavior of the suspect AS, a measurement campaign is carried out from the selected measurement ASes. This strategy is a building block for the complete TD location solution proposed next in Section 7.

Let i be the suspect AS. The proposed strategy searches for a pair of measurement ASes $W = (u, v)$, $u, v \in D$ that has not been previously selected, and for which all valley-free paths $p \in V^{\sigma}_{u,v}$ traverse i, i.e., $\forall p \in V^{\sigma}_{u,v}, i \in p$. The rationale is that if the suspect AS is in all possible paths between the selected measurement ASes, then it is guaranteed that the traffic from TD detectors will traverse the suspect AS, which may contribute to eliminating or confirming the suspicions about that AS when measurements are combined by Algorithm 1.

The search for AS pair W is limited by a parameter δ, which sets the maximum valley-free distance from i up to the limit at which measurement ASes are to be checked. Therefore, the proposed strategy tries to form an AS pair W starting from the measurement ASes closer to i, up to the measurement ASes that are at distance δ to i. The valley-free property improves the efficiency of this search, since it reduces the number of paths to check. For instance, if i itself is available for measurement ($i \in D$), then measurement pairs (i, j) are employed, such that the distance from i to j varies from 1 to δ. But if $i \notin D$, then measurement pairs are formed with ASes that are at distance 1 from i. In case those are also not available, pairs of ASes that are at a distance 2 from i are tried, and so on, up to distance δ.

Fig. 4 shows an example using the same portion of the graph as in Fig. 2. In this example, $\delta = 2$. There are four measurement ASes within distance 2 of i (the suspect AS to be investigated): $m_1, m_2, m_3, m_4 \in D$. The pair (m_1, m_2) follows the criteria described above and could be selected for investigating i, since all possible paths between m_1 and m_2 traverses i. However, pair (m_3, m_4) would not be selected, since there is a possible path between them that does not traverse i, which is $\{m_3, c_4, e_2, m_4\}$.

7. Complete solution for locating TD

In this section, we describe the complete solution proposed for locating TD in the Internet. This solution uses both the algorithm proposed in Section 5 and the strategy proposed in Section 6 to identify which AS is discriminating traffic between a given pair of end-hosts. The main idea is to filter out the neutral ASes from a list of suspects until only the discriminatory AS remains. The proposed solution receives as input a pair of ASes $E = (e_1, e_2)$, $e_1, e_2 \in A$, which we call the *initial pair*. The goal is to locate which AS in the paths between e_1 and e_2 is discriminatory. The output of the solution is a tuple (N, C, U), in which N is the set of neutral ASes, C the set of discriminatory ASes, and U the set of unknown ASes.

The proposed solution investigates each AS in the possible paths between the initial pair, filtering out the neutral ASes until only the discriminatory AS remains, by a process of elimination. The solution is divided in 5 steps, shown in Fig. 5. The Initialization step builds a set of suspect ASes (the ASes to be investigated). Then, in the AS Pair Selection step, two ASes are selected from the set of measurements ASes. Measurements are executed between the selected pair in the TD Detection step. The outcomes of these measurements, together with all previous measurements, are combined in the Inference step. The AS Pair Selection, TD Detection, and Inference steps are repeated until a halting condition is met. The solution finishes in the Completion step, returning the output. We describe each step in more detail next.

Initialization

In this step, a set of suspect ASes $S = A_{e_1,e_2}$ is selected, which consists initially of all the ASes in the valley-free paths between the initial pair E. The behavior of all ASes in S is initialized as unknown. Fig. 2 shows an example of an initial pair $E = (e_1, e_2)$ and all the possible valley-free paths $V^{\sigma}_{e_1,e_2}$, $\sigma = 0$.

AS pair selection

This step consists of choosing a suspect AS and using the strategy proposed in Section 6 to select a pair of measurement ASes W to investigate the chosen suspect. We take all discriminatory pairs $W' = (u, v)$, $u, v \in D$, $I_{u,v} = discriminatory$ for which measurements have already been taken in the TD detection step. The first time this step is executed, if the initial pair $E = (e_1, e_2)$ is available for measurement (i.e. $E \subseteq D$) then E is selected. Then, we count how many times each suspect $i \in S$ is present in all possible paths $V^{\sigma}_{u,v}$. The suspect i that appears less times is selected to be investigated. This heuristic relies on the fact that AS i is less likely to be discriminatory, and thus might be filtered earlier. If no discriminatory pair has been found yet, the first suspect in S is selected.

TD detection

In this step, the presence of TD between the pair of measurement ASes $W = (u, v)$, $u, v \in D$, selected in the previous step, is assessed. A TD detector is executed on u and v, and returns $I_{u,v}$.

Inference

This step consists of combining all measurements made by TD detectors using Algorithm 1. The input set M consists of all measurements obtained so far. The set of suspects S is updated using the output of the algorithm: neutral ASes are removed, and unknown ASes are added.

Completion

The halting conditions of the proposed solution for locating TD are the following: (i) an AS in the paths between the initial pair (A_{e_1,e_2}) is classified as discriminatory, thus TD has been located; (ii) all ASes in A_{e_1,e_2} are classified as neutral, thus no TD was found; and (iii) all measurement AS pairs have already been used to investigate the suspects — in this case TD could not be located, and one or more suspect ASes remain classified as unknown. If any of these conditions is met, the TD location solution finishes. The final output is tuple (N, C, U), i.e. the output of Algorithm 1 executed as part of the last Inference step.

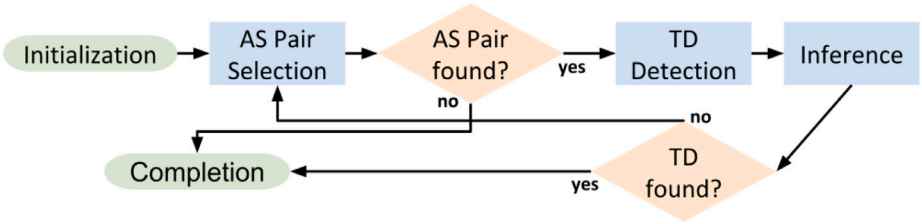


Fig. 5. Overview of the proposed solution for locating TD.

8. Evaluation: AS-level graph and paths

In this section we describe experiments for checking our assumptions related to Internet routing. The results shown in this section are based on an AS-level topology graph built using a dataset from the CAIDA research group, within their AS Rank project [28]. The same topology graph is also employed by the simulations presented in Section 9. Therefore, in this section we first describe the graph and the dataset from which it was built in Section 8.1. Next, the experiments for validating our assumptions are described in Section 8.2.

8.1. AS-level topology graph

The CAIDA dataset used to built the AS-level topology graph we employ in this work contains relationships between ASes in the Internet, inferred based on BGP data [26]. The dataset we used includes 86,622 unique ASes. 24,815 of these ASes have no relationship with other ASes, and were thus ignored in our evaluations. Therefore, we built a topology graph containing 61,807 different ASes. The graph $G = (A, L, f)$ was built by creating a vertex for each AS (set A), and an edge between each pair of ASes with a relationship in the dataset (set L). The type of relationship ($p2p$, $c2p$, $p2c$, or $s2s$) is indicated by a label on each edge (function f).

In this work, we employ two centrality metrics extracted from the topology graph: the betweenness and the valley-free betweenness. The betweenness centrality measures to which extent a vertex is present in the shortest paths between all other pairs of vertices. To be precise, the betweenness of a vertex is the sum of the fractions of shortest paths between all other pairs of vertices in which the vertex is present [35]. We call valley-free betweenness centrality a variation of this metrics that takes into account only the shortest valley-free paths. The strategy we propose for selecting measurement ASes rely on finding paths that traverse specific ASes, the suspects. Therefore, these metrics may be a good indication of the ability of ASes to be employed as measurement points. For instance, ASes with higher betweenness are present in more paths, which may turn them more likely to be selected for measurement.

8.2. AS-level paths in the internet

We conducted an experiment on the PlanetLab global testbed with three goals: (i) to assess to which extent the valley-free property is valid in the Internet; (ii) to determine the length of AS-level paths in the Internet; and (iii) to assess to which extent *traceroute* is a reliable tool for obtaining paths in the Internet.

In this experiment, we obtained the paths between numerous Internet IP prefixes and 29 PlanetLab hosts. The list of Internet prefixes employed, along with the corresponding ASes, was produced by CAIDA [36]. we chose a single prefix for the ASes with multiple prefixes. There were also a few ASes in the list of prefixes that do not appear in the AS-level topology graph we built. Such ASes were then discarded. In the end we employed 60,578 prefix/AS pairs in the experiment.

The paths to all prefixes/ASes were continuously measured from each PlanetLab host using the *traceroute* tool, from January 10, 2019 to February 1, 2019, for a total of 22 days of measurements. Each measurement resulted in a list of IP addresses, from a PlanetLab host to an Internet prefix, i.e. a host-level path. We converted all host-level paths to AS-level paths by mapping the IP addresses to the corresponding ASes. This mapping was performed using the same list of prefixes from CAIDA found in [36]. However, a common issue with traceroute measurements is that some hosts along the path do not send a reply after the probes, or reply with an invalid address. In these cases, we may not know that the corresponding AS is in the path, unless another host in the same AS replies to the probe. We describe how we addressed this issue below.

The AS-level paths obtained were then classified as *valley*, *valley-free*, or *unknown*. Valley-free paths present the valley-free property in the topology graph, as described in Section 3, while valley paths do not present the property. For the paths that presented measurement errors, such as described above, we first checked if they presented the valley-free property when ignoring the errors. In these cases we classified the paths as valley-free, assuming that another host in the same AS replied to the traceroute probes. Otherwise, we classified the paths as unknown, since we cannot know if the measured path is complete (there might be ASes missing in the obtained AS-level path) and thus cannot know the actual classification. There were also a few paths containing links not present in the graph, which were excluded from our results.

A total of 75,597,104 *traceroute* measurements were collected, out of which 1,801,089 (2.38%) had links not present in the graph and were thus excluded, resulting in 73,796,015 AS-level paths. A total of 40,837,151 unknown paths were observed (55.34%). The remaining paths consisted of 32,703,036 (44.31%) valley-free paths, and 55,828 (0.35%) valley paths. A total of 48,283 unique ASes were reached through the valley-free paths (79.7% of all prefixes measured).

We also investigated parameter σ, to discover how frequently and by how much AS-level paths are larger than the shortest paths. For each of the 32,703,036 valley-free paths measured, we compared its length with the length of the shortest valley-free path between the same pair of ASes in the graph. Results show that 55.78% of the paths measured had the same length as the shortest path in the graph, 31.87% were one link larger, and 10.34% were two links larger.

Results show that the vast majority of paths that were successfully measured (i.e. they are not unknown) followed the valley-free property, which is a key assumption of the present work. On the other hand, more than half of the measurements resulted in unknown paths, which shows the limitations of the traceroute-like techniques, which are employed by other existing proposals for locating TD as described in Section 2. Finally, the majority of the observed valley-free AS-level paths in the Internet (87.65%) has length at most a single link larger than the corresponding shortest path.

9. Evaluation: Locating TD

In this section, we present simulation results executed to evaluate the complete solution for locating TD proposed in Section 7. The main goals of the experiments are: (i) to evaluate whether the proposed solution is capable of locating TD under different scenarios; (ii) to evaluate how measurement points with different characteristics impact

the efficiency of the proposed solution; and (iii) to identify between which pairs of ASes it is more efficient to locate TD with the proposed solution.

The rest of this section is organized as follows. We first describe the methodology in Section 9.1. Then we give details about the implementations in Section 9.2. Next, the simulation scenarios are presented in Section 9.3. We then describe the parameters employed in the simulations, and how we chose their values in Section 9.4. Section 9.5 presents results comparing several different sets of measurement ASes, while Section 9.6 compares different sets of initial pairs. We then present results based on different assumptions: in Section 9.7 do not assume that the initial pairs are available for measurement, and in Section 9.8 we consider paths larger than the shortest paths. Finally, we discuss the results and limitations of our evaluation in Section 9.9.

9.1. Simulation roadmap

We evaluated the complete solution for locating TD described in Section 7 under several different scenarios, varying the initial pair of end-hosts E between which TD is to be located, the set of measurement ASes D, as well as parameter σ. Results are evaluated according to three criteria: (i) the success rate, i.e. the percentage of simulations that located TD successfully in each scenario; (ii) the average number of measurement AS pairs selected by the solution across all simulations for each scenario, i.e. the average number of measurements; and (iii) the number of measurement ASes available that can be selected in each scenario, i.e. the size of set D.

The *optimal* set of measurement ASes D is the one that achieves the highest success rate, issuing the smallest number of measurements, and containing the smallest number of ASes available for measurement. The rationale is that the number of ASes available for measurement may be limited. Furthermore, issuing a large number of measurement campaigns imposes an overhead on the network.

We do not compare our solution with related work, since as described in Section 2 our solution relies on different assumptions and thus addresses a different problem. We do not claim that our solution achieves better success rates or requires less measurements than other existing solutions. We do claim that our proposals rely on more realistic assumptions with respect to path knowledge.

9.2. Implementation

We implemented the proposed solution in C++, using the Boost Graph Library.[1] The implementation followed a modular design, allowing any TD detector to be used as a module of the software. For the purpose of evaluation, in addition to δ and σ, we added another two parameters to the implementation, mt and mp. Parameter mt is an upper bound for the number of AS pairs that are checked to investigate a suspect. For instance, when searching for an AS pair to investigate a suspect AS s, if for mt different AS pairs the paths between them do not all traverse s, the strategy no longer tries to investigate s. Parameter mp is an upper bound on the number of AS pairs that may be selected to investigate a given suspect. Thus, after mp AS pairs have been selected to investigate a suspect, no more pairs will be selected for that suspect. The goal of these parameters is to limit the search space with respect to measurement AS pairs, in order to make it feasible to run a large number of simulations. We describe how the values of these parameters were set in Section 9.4.

The simulator itself was also implemented in C++, and executes the solution for locating TD under different scenarios, and uses an "oracle" as the TD detector. The oracle receives as input two ASes $u, v \in A$, and returns the inferred behavior $I_{u,v}$. The oracle has perfect knowledge about which AS k is discriminatory. The oracle checks if AS k is in any valley-free path between u and v, i.e. if $k \in A_{u,v}$. If no, then the oracle returns $I_{u,v} = neutral$. Otherwise, $I_{u,v} = discriminatory$. The oracle considers that if k is in at least one path between u and v, then that would be the path traversed by traffic, i.e. it always assumes the worst case. The rationale for not using a real TD detector is that our goal is to evaluate the proposals for locating TD, which rely on any type of TD detector.

All simulations presented in this section were executed on a server machine based on an Intel Xeon E5-2690 v2 processor with 200GB of RAM memory, running Linux Mint 19.1.

[1] https://www.boost.org/doc/libs/1_76_0/libs/graph/doc/index.html

Table 2
Selected sets of ASes.

Name	Description	Size
pdb-access	Access providers from PeeringDB	5263
pdb-content	Content providers from PeeringDB	1462
pdb-transit	Transit providers from PeeringDB	2293
degree-eq-1	ASes with degree 1 in the graph	21220
degree-le-2	ASes with degree ≤ 2 in the graph	41247
degree-top-n	ASes with the highest degree	n
vfbet-top-n	ASes with the highest valley-free betweenness centrality	n
bet-top-n	ASes with the highest betweenness centrality	n

9.3. Simulation scenarios

The simulator executes the solution for locating TD under multiple scenarios. Each simulation scenario receives as input set $Z = \{(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)\}$, $u_i, v_j \in A$ of initial pairs, and a set $D \subseteq A$ of measurement ASes. In each experiment and for each scenario several simulations are executed. For each initial pair $E = (e_1, e_2)$, we take each AS $k \in A_{e_1,e_2}$ (the ASes in the paths between e_1 and e_2) and execute a simulation in which k is the AS responsible for TD. The simulation is considered to be successful if AS k is classified as discriminatory. Furthermore, we also execute a simulation with no AS employing TD, in which case the simulation is successful if all ASes $u \in A_{e_1,e_2}$ are classified as neutral. Therefore, each scenario results in $\sum_{E \in Z}(|A_{e_1,e_2}| + 1)$ simulation runs.

All simulations were executed on the same AS-level topology graph G, built from the CAIDA dataset, as described in Section 8. We assume at first that on each simulation the ASes in E are also available for measurement, in addition to those ASes in set D. We also present results without this assumption in Section 9.7.

Sets Z and D were built based on metrics extracted from the graph, as well as on the classification of ASes available on the PeeringDB website [37]. PeeringDB is an online database in which operators share information regarding their networks. According to [38], the number of ASes registered on the website as transit, access and content providers is representative of the corresponding sets in the Internet. We obtained the list of ASes of these types from PeeringDB in June 20th, 2019. Furthermore, we ordered the ASes based on degree, betweenness centrality, and betweenness centrality taking into account only valley-free paths. Table 2 shows the sets of ASes employed. The columns of the table indicate for each set: name, description, and number of ASes. The first three sets were taken from the PeeringDB website. The last three sets consist of the n ASes with the highest values for the corresponding metrics. The values of n we employed were: 10, 50, 100, 500, and 1000.

We created six sets of initial pairs, shown in Table 3, using the sets of ASes described above. Each of these sets contains 1000 different pairs of ASes. The table also shows the total number of simulations executed on scenarios employing each set. Set *pdb-a2a* contains 1000 pairs randomly selected from the ASes in the *pdb-access* set, i.e., from all possible pairs between access providers (from PeeringDB), we randomly picked 1000 pairs. This set represents a common situation in the Internet: two end-hosts, connected to access providers, communicating with each other, such as in a P2P application. Analogously, sets *pdb-c2c* and *pdb-t2t* are composed of ASes from sets *pdb-content* and *pdb-transit*,

Table 3
Sets of initial pairs.

Name	Pair Composition	Simulations
pdb-a2a	Access providers	7818
pdb-c2c	Content providers	7229
pdb-t2t	Transit providers	7168
pdb-a2c	Access and content providers	7807
pdb-a2t	Access and transit providers	7609
pdb-c2t	Content and transit providers	7295

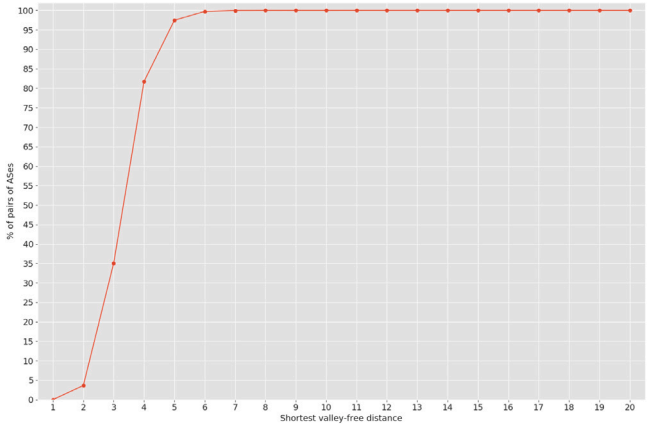


Fig. 6. CDF of the valley-free distances.

respectively. Moreover, the *pdb-a2c* set contains 1000 pairs randomly selected in such a way that one of the ASes in each pair is from the *pdb-access* set and the other from the *pdb-content* set. This represents another common situation: an end-user accessing a content provider, such as a video streaming service. Similarly, sets *pdb-a2t* and *pdb-c2t* are composed of access/transit providers and content/transit providers, respectively.

9.4. Parameters

Our proposals employ two parameters, δ and σ. We employed $\sigma = 0$ in most experiments presented in this section, thus we examine only the shortest valley-free paths between ASes. We do, however, present results for $\sigma = 1$ in Section 9.8, since paths one link larger than the shortest path are common in the Internet, according to the experiments described in Section 8. As for parameter δ, we set $\delta = 2$ on all simulations, thus only measurement ASes up to 2 hops away from the suspects are considered. Higher values would significantly increase the search space, since a large portion of the graph would be at a distance of 3 or more hops from the suspects. Furthermore, as we observed in the results presented later in this section, measurement ASes farther from the suspects are rarely selected. Fig. 6 shows the Cumulative Distribution Function (CDF) of the valley-free distances for all pairs of ASes in the graph. The figure shows for each distance value the rate of pairs of ASes distant to each other up to that value. For instance, about 5% of all pairs of ASes are up to 2 hops away from each other. On the other hand, for a distance of up to 3 hops, the rate raises to about 35% of the AS pairs.

In order to choose values for parameters *mp* and *mt*, we ran several simulations employing different values. In these simulations, we employed a set of initial pairs Z containing 1000 pairs selected randomly from all ASes in the graph. We employed two different sets of ASes as the measurement ASes D: *degree-le-2* and *vfbet-top-1000*. These two sets presented the best results overall, as described later in this section.

First, we ran several sets of simulations employing a fixed large value for *mp*, and several different values for *mt*. We employed $mp = 100$, while *mt* ranged from 10 to 100, in increments of 10. For each value of *mt*, 8479 simulations were executed. Fig. 7 shows the results obtained from these simulations. The success rate achieved by each set of measurement ASes for each value of *mt* is shown in Fig. 7(a), while Fig. 7(b) shows the average number of probes issued when using each set and for each value of *mt*. It is possible to see that both the success rate and the average number of probes did not vary much as the value of *mt* increased. We chose $mt = 20$ for our simulations, which is the value for which the success rate had the largest increment for both sets of measurement ASes. Therefore, we discard a measurement AS after 20 attempts for each suspect.

Next, we ran simulations with $mt = 100$, and *mp* ranging from 10 to 100. Fig. 8 shows the results obtained from these simulations. The success rate achieved by each set of measurement ASes for each value of *mp* is shown in Fig. 8(a), while Fig. 8(b) shows the average number of probes issued when using each set and for each value of *mp*. The success rate and average number of probes for the *vfbet-top-1000* did not increase much as the value of *mp* increased. However, for the *degree-le-2* set, both the success rate and the number of average probes increased significantly. We chose $mp = 40$, since larger values would significantly increase the search space, and thus also the execution times, but without achieving significantly better results. Therefore, in our simulations, up to 40 AS pairs are selected for each suspect.

9.5. Results: Comparing measurement ASes

We first present results comparing the following metrics: degree, betweenness and valley-free betweenness. The sets of measurements ASes *degree-top-n*, *bet-top-n* and *vfbet-top-n*, for $n \in \{10, 50, 100, 500, 1000\}$, were built based on these metrics, respectively. The success rate achieved by each of these sets is shown in Fig. 9, on scenarios employing $Z = pdb\text{-}a2a$. For all values of n, the highest success rates were achieved by sets *vfbet-top-n* (from 29% for *vfbet-top-10* to 93% for *vfbet-top-1000*), and the lowest by sets *degree-top-n*. Given the best success rates were achieved by *vfbet-top-1000*, we will not show results for the other sets in the remainder of this work.

The distance between ASes in *vfbet-top-n* and suspects is generally smaller, in comparison with ASes in *degree-top-n* and *bet-top-n*. For instance, the average valley-free distance from ASes in *vfbet-top-1000* to suspects was 0.79, while it was 0.87 for *degree-top-1000* and 0.85 for *bet-top-1000*. Being closer, there are less paths and less ASes between pairs of ASes from *vfbet-top-1000*, and thus it is less likely that the discriminatory AS k is present on the measurements between them, which results in suspects being filtered earlier (inferred as neutral).

We evaluate next the sets of measurement ASes (D) *degree-eq-1, degree-le-2, pdb-access, pdb-transit*, and *vfbet-top-1000*. Results for these sets are shown in Fig. 10, with $Z = pdb\text{-}a2a$. The success rates of each set D across (i) all simulations, (ii) simulations in which k was in the initial pair E, and (iii) simulations in which k was not in E are shown in Fig. 10(a). The average number of probes (i.e. requests to the oracle) across (i) all simulations, (ii) successful simulations, and (iii) unsuccessful simulations are shown in Fig. 10(b). Beside each bar there are two values, one indicating the total number of unique ASes that were selected for measurements across all simulations for each set D, and another indicating the total number of ASes available for measurement ($|D|$).

Sets *degree-le-2* and *vfbet-top-1000* achieved the highest success rates, 94% and 93%, respectively. However, simulations employing *degree-le-2* issued significantly more probes on average. The distance between ASes in *degree-le-2* is usually larger, which makes k more likely to be within the paths between those ASes, causing more pairs to be selected in order to find neutral ASes. The average valley-free distance between ASes from *degree-le-2* was 2.01, and 1.48 between ASes from *vfbet-top-1000*. Similarly, the average distances to the suspects were 1.8 and 0.79, respectively. Regarding the simulations in which no TD was present, the success rates achieved by *degree-le-2* and *vfbet-top-1000* were 94% and 91%, while the average numbers of probes

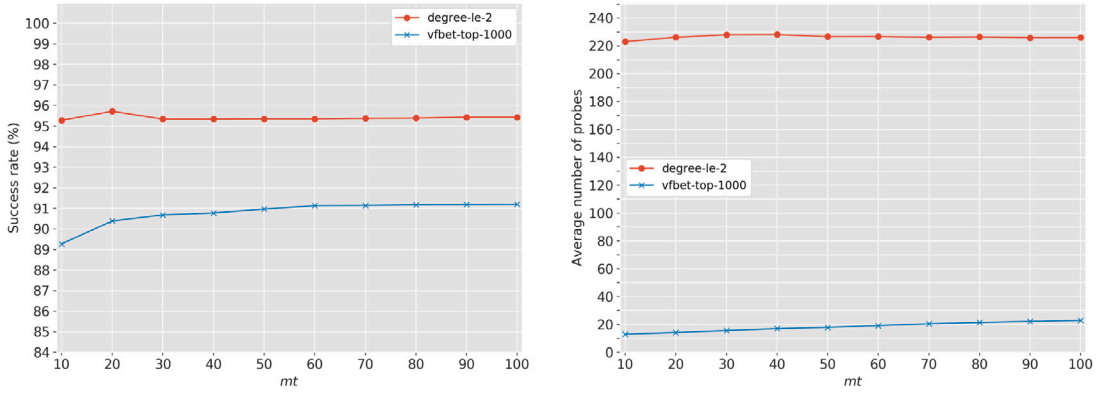


Fig. 7. Success rates and average probes achieved by the sets of measurement ASes *degree-le-2* and *vfbet-top-1000*, for $mp = 100$ and mt ranging from 10 to 100.

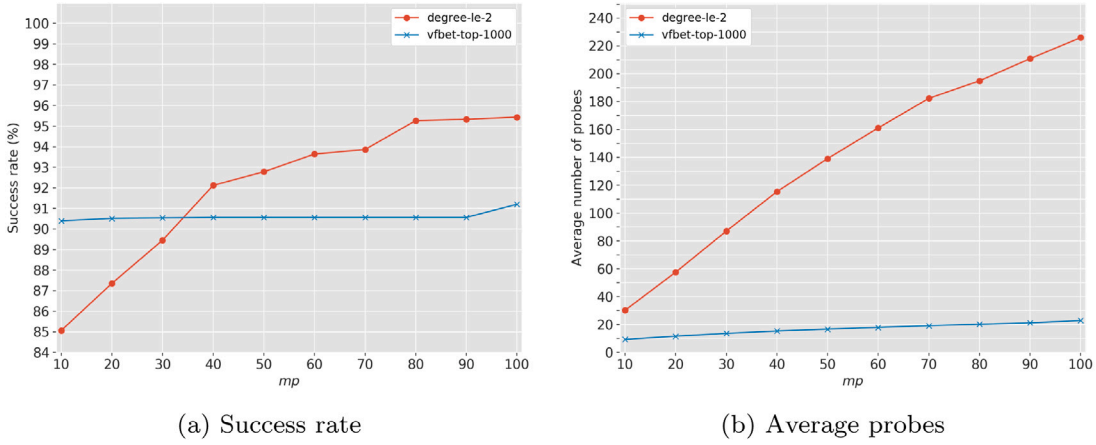


Fig. 8. Success rates and average probes achieved by the sets of measurement ASes *degree-le-2* and *vfbet-top-1000*, for $mt = 100$ and mp ranging from 10 to 100.

were 5.27 and 5.15, respectively. On these simulations, there was no discriminatory AS, thus the AS pairs selected always resulted in a suspect being filtered out.

From the 41,247 available ASes in *degree-le-2*, 8269 were selected for measurement on all simulations, while 615 (from a total of 1000) were selected from *vfbet-top-1000*. This shows that the *vfbet-top-1000* set achieved a similar result in terms of success rate using significantly less different ASes (615 *vs* 8269). As discussed later in this section, ASes in the core of the Internet are better positioned than those in the edge. Therefore, having access to a much smaller number of core ASes is enough (1000 *vs* 41,247). Note that the graph employed in our simulations has 61,807 ASes in total.

A slightly lower success rate, 88%, was observed for set *pdb-transit*, relative to *vfbet-top-1000*. The average number of probes was also similar, but a larger number of unique ASes were selected for measurement (811 from a total of 2293) for *pdb-transit*. The lowest success rates observed correspond to sets *degree-eq-1* (77%) and *pdb-access* (71%). 6271 ASes, out of 21,220 available, were selected from *degree-eq-1*, while 2177, out of 5263 available, were selected from *pdb-access*. Furthermore, set *degree-eq-1* issued significantly more probes than *pdb-access* on average, due to the same reasons described above for set *degree-le-2*. It is also possible to observe that the average number of probes on successful simulations is significantly lower than on unsuccessful simulations, for all sets of measurement ASes employed. This happens due to the halting conditions adopted by our strategy. On unsuccessful simulations, all possible pairs of measurement ASes are selected before the strategy finishes its execution.

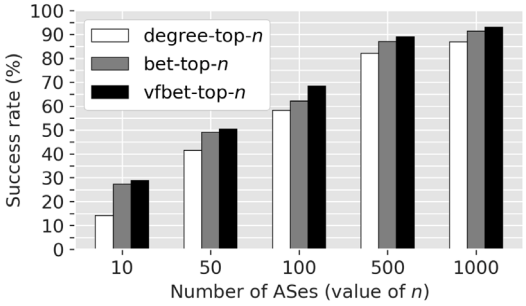
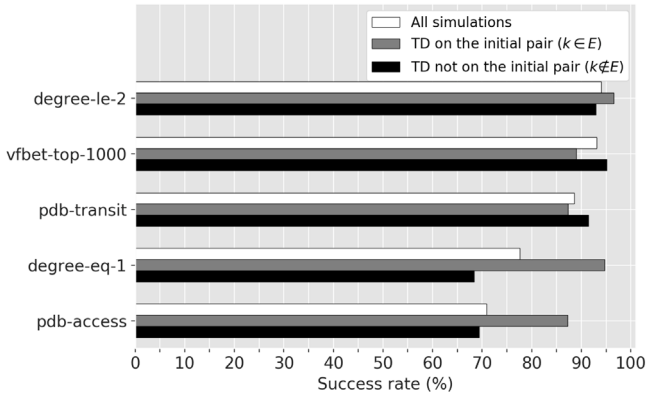


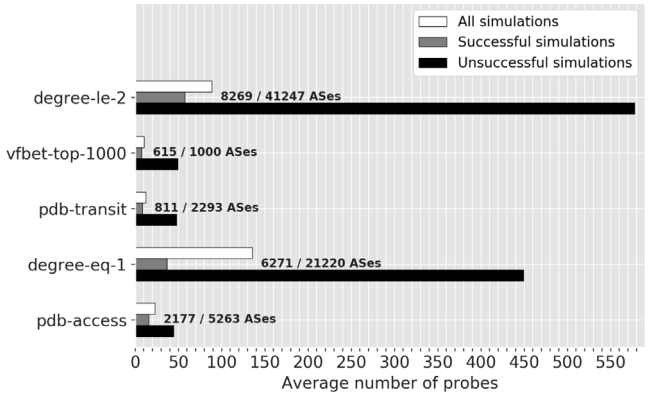
Fig. 9. Success rates for Z = *pdb-a2a*, and varying sizes of *degree-top-n*, *vfbet-top-n* and *bet-top-n* as D.

9.6. Results: Comparing initial pairs

We now present results comparing different sets of initial pairs (Z). We present results for sets of measurement ASes *degree-le-2* and *vfbet-top-1000*, which presented the highest success rates and contain ASes at different parts of the Internet — edge (*degree-le-2*) and core (*vfbet-top-1000*). Fig. 11 shows the success rates (for all simulations, $k \in E$, and $k \notin E$) for the sets of initial pairs *pdb-a2a, pdb-c2c, pdb-t2t, pdb-a2c, pdb-a2t*, and *pdb-c2t*. Fig. 11(a) shows the results for D = *vfbet-top-1000*, while Fig. 11(b) for D = *degree-le-2*. Furthermore, Fig. 12 shows the average number of probes for (i) all simulations, (ii) simulations that were successful, and (iii) unsuccessful simulations, for each set of

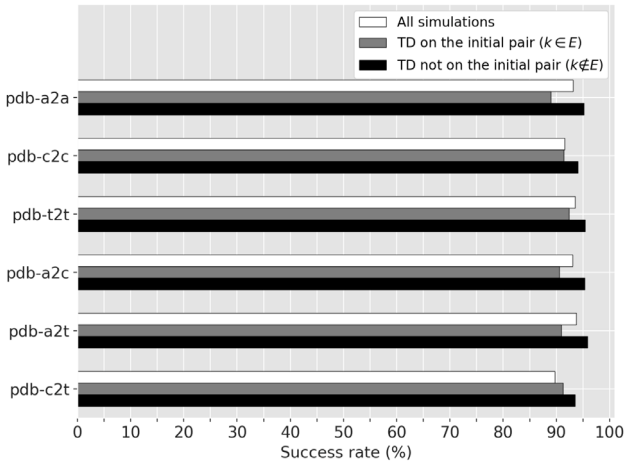


(a) Success rate

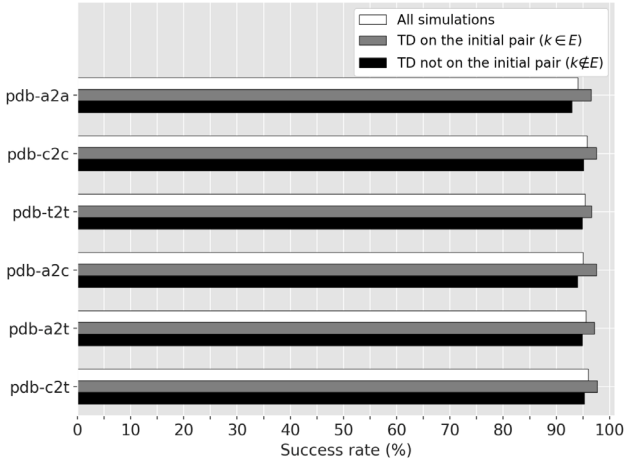


(b) Average probes

Fig. 10. Success rates and average probes for different sets D, and Z = *pdb-a2a*.



(a) D = vfbet-top-1000



(b) D = degree-le-2

Fig. 11. Success rates for different initial pair sets Z, with D = vfbet-top-1000 and D = degree-le-2.

initial pairs. Fig. 12(a) shows the results for D = *vfbet-top-1000*, while Fig. 12(b) for D = *degree-le-2*.

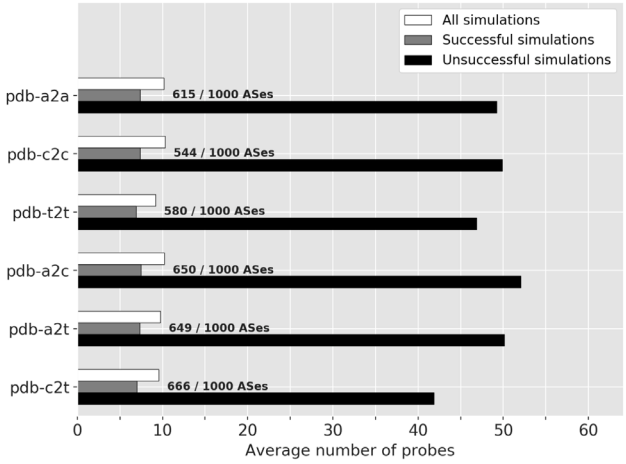
It is possible to conclude that both measurement sets had similar success rates for all sets of initial pairs. The success rates for *vfbet-top-1000* ranged from 89% to 93%, while the success rates for scenarios with *degree-le-2* ranged from 94% to 96%. The main difference between the two sets was that scenarios with *degree-le-2* employed significantly more probes on average, ranging from 73.12 to 102.48. The number of different ASes selected for measurement from *degree-le-2* ranged from 6756 to 9084 (from a total of 41,247). For scenarios with *vfbet-top-1000*, the average number of probes ranged from 9.16 to 10.28, and the number of ASes selected for measurement ranged from 544 to 666 (from a total of 1000).

9.7. Results: $E \not\subset D$

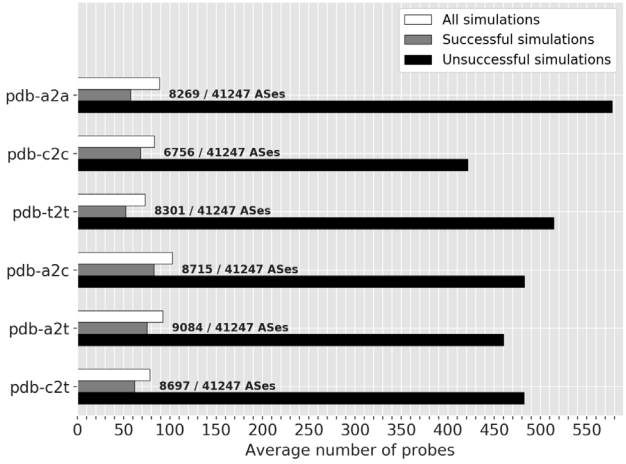
We also simulated scenarios considering that the initial pair E is *not* available for sending probes. The goal of this set of simulations is to check if it is possible to detect TD between two ASes that we do not have access to (in order to run TD detectors on). In these scenarios, only the ASes in D are available to be selected for measurement — and in case the ASes in the initial pair are present in D, we remove them from the set for that simulation scenario to ensure they are not available. Fig. 13 shows the success rates for the sets of measurement ASes *vfbet-top-1000* 13(a) and *degree-le-2* 13(b), on scenarios with different sets Z. Similarly, Fig. 14 shows the average number of probes for the sets of measurement ASes *vfbet-top-1000* 14(a) and *degree-le-2* 14(b).

Results show that the success rates for both sets of measurement ASes were similar. The success rates for all simulations on scenarios with *vfbet-top-1000* ranged from 49% to 56%, while the success rate for *degree-le-2* ranged from 50% to 57%. As expected, the success rates for both sets were significantly lower than those of the scenarios previously presented (for $E \subset D$). However, for both sets, the success rates for the simulations in which $k \notin E$ were significantly higher than for simulations with $k \in E$. For *vfbet-top-1000*, the success rates when $k \notin E$ ranged from 83% to 90%, and for *degree-le-2* ranged from 72% to 81%. When $k \in E$, the success rates ranged from 0% to 1% for *vfbet-top-1000*, and from 8% to 37% for *degree-le-2*. We explain these results below.

Due to the valley-free property, there may be no paths between ASes of the Internet core that traverse ASes in the edge of the Internet (or closer to the edge). ASes in the core, such as the ASes in *vfbet-top-1000*, are mostly connected to other ASes through $p2p$ or $p2c$ relationships — they are on the top of the Internet hierarchy (Tiers 1 and 2). For instance, only 1.8% of the relationships from ASes in *vfbet-top-1000* to other ASes are $c2p$. Therefore, the paths between these ASes usually consist of other ASes with the same characteristics. If a path between two such core ASes traverses an AS in the edge it would violate the valley-free property, since at some point there would be a $p2c$ link to the AS in the edge, followed by a $c2p$ link going back to an AS in the core – i.e., a "valley". In this set of simulations, since the initial pair of ASes is not available for measurement, our strategy needs at least one measurement pair for which the paths traverse the discriminatory

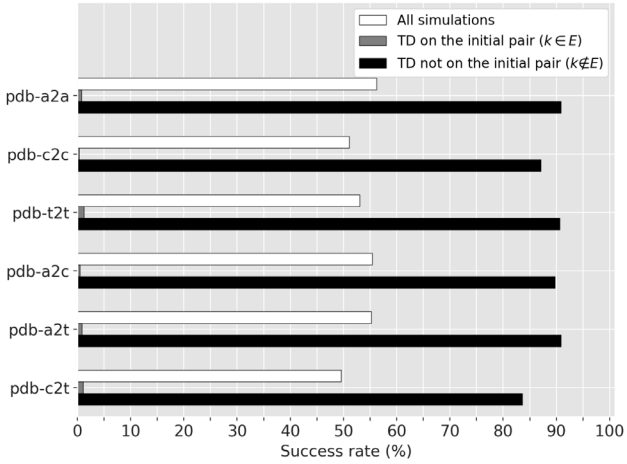


(a) D = vfbet-top-1000

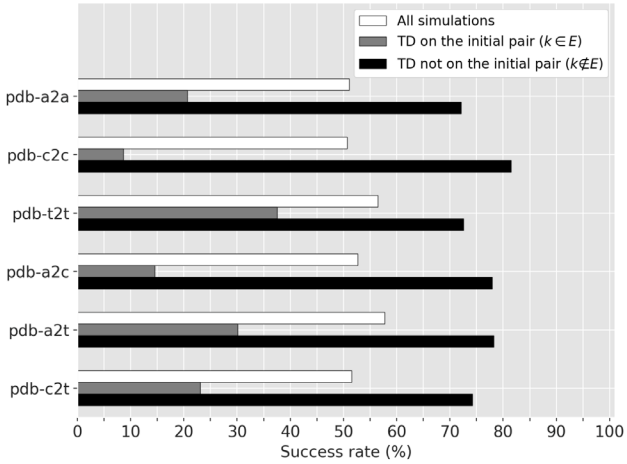


(b) D = degree-le-2

Fig. 12. Average number of probes for different initial pair sets Z, with D = vfbet-top-1000 and D = degree-le-2.



(a) D = vfbet-top-1000



(b) D = degree-le-2

Fig. 13. Success rates for different sets Z, with D = vfbet-top-1000 and D = degree-le-2, and $E \not\subset D$.

AS k: new suspects are then found, potentially better positioned so it is possible to measure and find them. However, it is often not possible to find such measurement pair when $k \in E$. The ASes in *degree-le-2* are in the edge of the Internet, so it was possible to find paths traversing some of the ASes in the initial pairs, hence the higher success rates.

Furthermore, the average number of probes in unsuccessful simulations was significantly lower for $E \not\subset D$, when compared to the results presented previously in this section. However, the average number of probes in successful simulations is similar. For instance, let us take D = *vfbet-top-1000* and Z = *pdb-a2a*. The average number of probes in successful simulations for this configuration and $E \subset D$ was 7.39, as can be observed in Fig. 12, while in unsuccessful simulations the average was 49.22. For $E \not\subset D$ (Fig. 14), the average in successful simulations was 5.36, while the average in unsuccessful simulations was 13.72. The reason for this behavior is the same as described above: in the unsuccessful simulations, our strategy was able to find a much lower number of suitable AS pairs for issuing probes from when $E \not\subset D$. In the successful cases, a similar number of AS pairs was necessary.

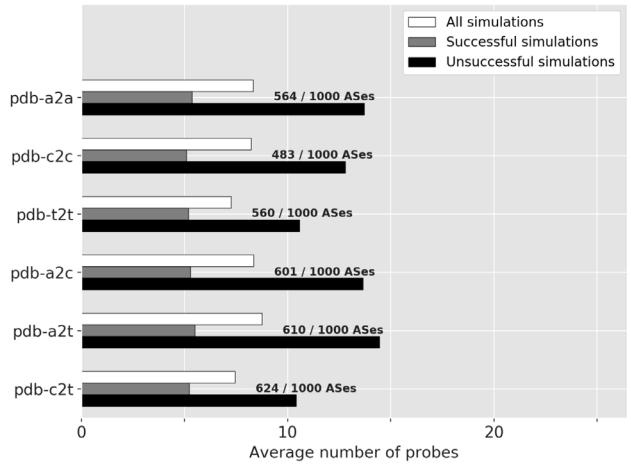
9.8. Results: $\sigma = 1$

In the experiments previously described in Section 8, 55.78% of the measured valley-free paths had the same size of the corresponding shortest valley-free path in the graph, while 31.87% of the measured valley-free paths were one link larger than the shortest path. These represent 87.65% of all valley-free paths observed in the experiments. Therefore, we executed several simulations with $\sigma = 1$ to check if our proposal is capable of locating TD with a larger number of possible paths between end-hosts.

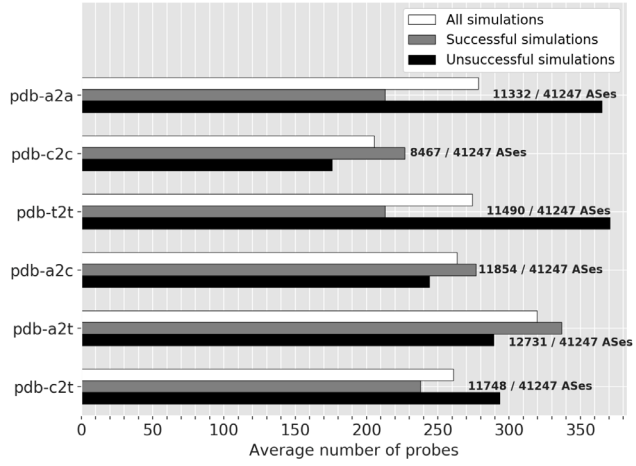
Fig. 15 shows the success rates for the sets of measurement ASes *vfbet-top-1000* 15(a) and *degree-le-2* 15(b), on scenarios with different sets Z. Similarly, Fig. 16 shows the average number of probes for the sets of measurement ASes *vfbet-top-1000* 16(a) and *degree-le-2* 16(b).

For the *vfbet-top-1000* set, the success rates ranged from 88% to 90% for all simulations. These values were similar to the success rates for $\sigma = 0$ (Fig. 11), which ranged from 89% to 93%. For the *degree-le-2* set, the success rates for $\sigma = 1$ ranged from 84% to 87%. For $\sigma = 0$, the success rates ranged from 94% to 96% (Fig. 11). It is also possible to observe that the average number of probes increased significantly for both sets in comparison with the results presented previously in Fig. 12.

In all results presented in previous subsections, the success rates were always slightly higher for *degree-le-2*. However, for $\sigma = 1$, the success rates are slightly higher for the *vfbet-top-1000* set. Since ASes in *vfbet-top-1000* are generally closer to each other, the number of possible paths between measurement ASes increases much more for *degree-le-2* then for the *vfbet-top-1000* set when $\sigma = 1$.

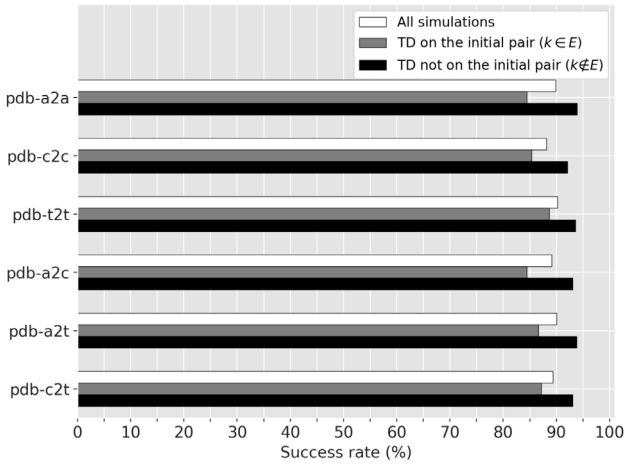


(a) D = vfbet-top-1000

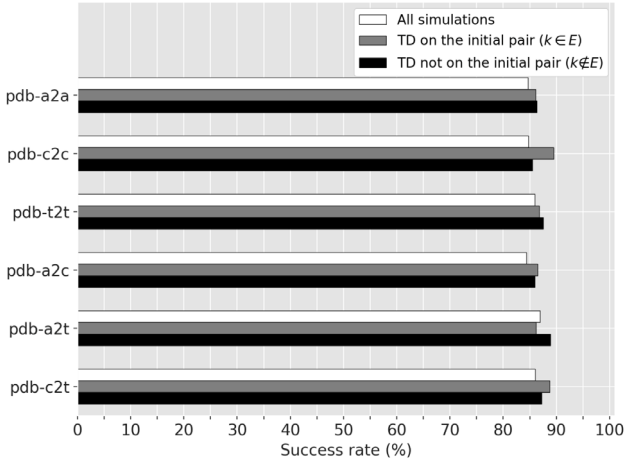


(b) D = degree-le-2

Fig. 14. Average number of probes for different sets Z, with D = vfbet-top-1000 and D = degree-le-2, and $E \not\subset D$.



(a) D = vfbet-top-1000



(b) D = degree-le-2

Fig. 15. Success rates for different initial pair sets Z and $\sigma = 1$, with D = vfbet-top-1000 and D = degree-le-2.

We also present results for $\sigma = 1$ and $E \not\subset D$, i.e., not considering that the initial pair is available for measurement. Fig. 17 shows the success rates for the sets of measurement ASes *vfbet-top-1000* 17(a) and *degree-le-2* 17(b), on scenarios with different sets Z and $E \not\subset D$.

Surprisingly, the success rates for both sets of measurement ASes were higher than those of the results presented previously in Fig. 13 (for $\sigma = 0$). The success rates of all simulations for *vfbet-top-1000* ranged from 73% to 75% (49% to 56% in previous results). For the *degree-le-2* set, the success rates ranged from 62% to 68% (50% to 57% in previous results). The reason for this behavior is that there are more possible paths between measurement ASes with $\sigma = 1$, thus it is easier to find a measurement pair for which the paths contain the discriminatory AS k. When such pair is found, new suspects start to be investigated, which are better positioned (relative to the valley-free property) than the initial suspects, as we explained previously in Section 9.7: when $\sigma = 0$, it is less likely that k will be in the paths between the measurement ASes.

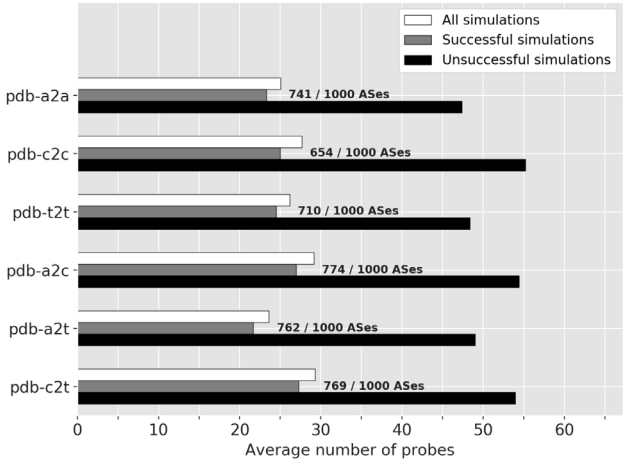
9.9. Discussion

The results presented in this section show that the proposed solution is capable of inferring the behavior of ASes by combining inferences from multiple TD detectors running on different ASes. Table 4 shows a summary of the different simulation scenarios and the main conclusions that can be drawn from the results. Selecting measurement points based on the valley-free betweenness centrality presented good results in our simulations. Another finding was that having a large number of ASes available for measurement in the edge of the Internet (41,247 from the *degree-le-2* set) achieved similar success rates than having access to only a few ASes in the core (1000 from the *vfbet-top-1000* set). Furthermore, much less probes were issued when employing core ASes, since they are usually closer to a larger portion of the network, compared to ASes in the edge. Thus, in order to achieve higher success rates, a much larger number of edge ASes may be necessary at multiple portions of the network, covering several vantage points. Finally, results show that it is possible to locate TD between any two ASes, even if we do not have access to them for issuing probes. However, locating TD that is happening in the core of the Internet is easier then locating TD in the edge.

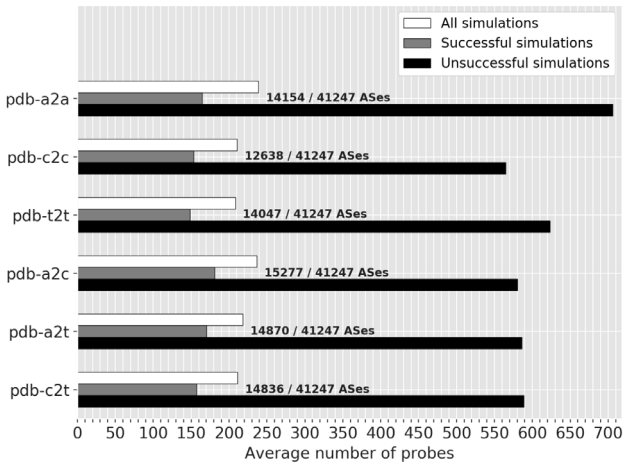
We highlight that our simulations never produced false-negatives (a discriminatory AS k was never inferred as neutral) nor false positive results (neutral ASes were never inferred as discriminatory). However, some of our assumptions may not always be true in the wild. Traffic may traverse valley paths, ASes may differentiate traffic based on their origin or destination, and the real AS-level topology may be slightly different. In such cases, the proposed solution for locating TD may

Table 4
Evaluation summary.

Scenario	Section	Figures	Results
Graph metrics	9.5	9	Valley-free betweenness centrality presented the best results among the metrics extracted from the graph
Comparing sets of measurement ASes	9.5	10	Sets *degree-le-2* and *vfbet-top-1000* achieved the highest success rates, but *degree-le-2* issued significantly more probes on average
Comparing sets of initial pairs	9.6	11, 12	Similar success rates were observed for all sets of initial pairs
Initial pairs not available as measurement ASes ($E \not\subset D$)	9.7	13, 14	Success rates were significantly higher when the discriminatory AS is not in the initial pair
Considering longer paths ($\sigma = 1$)	9.8	15,16	Success rates were similar to previous results, but the number of probes were significantly higher
Considering longer paths and that initial pairs are not available as measurement ASes	9.8	17	Surprisingly, success rates were higher than in Fig. 13

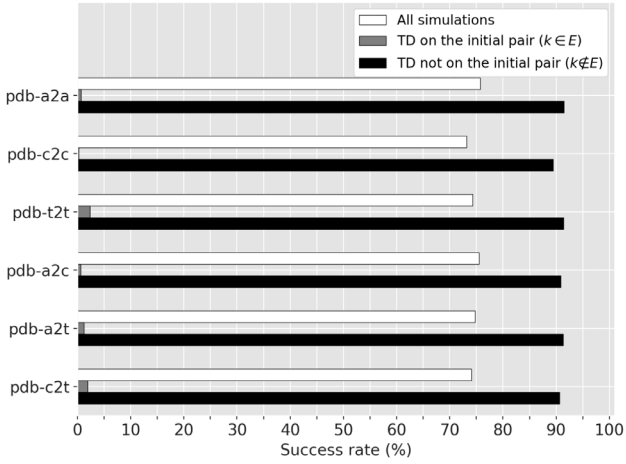


(a) D = vfbet-top-1000

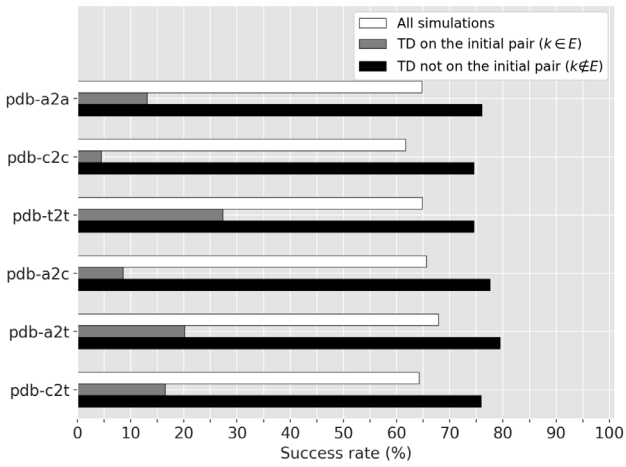


(b) D = degree-le-2

Fig. 16. Average number of probes for different initial pair sets Z and $\sigma = 1$, with D = vfbet-top-1000 and D = degree-le-2.



(a) D = vfbet-top-1000



(b) D = degree-le-2

Fig. 17. Success rates for different initial pair sets Z and $\sigma = 1$, with D = vfbet-top-1000 and D = degree-le-2, and $E \not\subset D$.

result in false-positives or false-negatives. It is also worth noticing that in our simulations, the oracle always assumed the worst case, i.e., it considered traffic would always follow the path containing the discriminatory AS k. However, in a real situation the actual path may not traverse k, in which case fewer probes might be necessary to locate TD, since suspects might be filtered earlier.

In the case of unsuccessful simulations, we did not evaluate how close the solution was to locating TD. Even when the solution is not able to find exactly which AS was discriminatory, the set of suspects may have been filtered to just a few, which can be helpful. Another limitation of our evaluations is that in each simulation we considered that only a single AS was discriminatory. Although in real conditions this may not be true, a more controlled environment was employed in

order to (i) evaluate if the proposed solution was actually capable of locating TD in the first place, (ii) to identify the trade-offs between success rate and number of measurements, and (iii) to identify the relations between our proposals, the valley-free property, and different types of measurement ASes and initial pairs. Finally, parameters *mt* and *mp* limited the number of measurements that could have been issued during the simulations. Without these parameters, higher success rates could be achieved, but at the expense of more measurements.

10. Conclusion

In this work we addressed the problem of locating TD in the Internet under more realistic assumptions than existing strategies. A solution for locating the exact AS that is discriminating traffic is proposed. The solution considers all possible AS-level valley-free paths, instead of relying on the exact knowledge of host-level paths (as other existing strategies do). The solution consists of an algorithm for combining measurements from TD detectors running on different ASes plus a strategy for selecting the best ASes to run measurements.

To evaluate our proposals, we first conducted a series of experiments on PlanetLab, in which we executed *traceroute* a large number of times for determining the paths from a set of end-hosts to several Internet prefixes. Results show that *traceroute*-like techniques employed by other solutions for locating TD may not be reliable, and that the vast majority of the paths that were successfully measured do follow the valley-free property. We then executed simulations for evaluating the proposed solution for locating TD. We defined several scenarios, varying the location of TD and the measurement points employed. We draw four main conclusions from the results obtained on these simulations: (i) few measurement ASes in the core of the network achieve similar results as a much larger number of measurement ASes in the edge; (ii) it is possible to locate TD between any two ASes in the Internet, even if they are not accessible for issuing probes from; (iii) due to the valley-free property, it is easier to locate TD in the core than in the edge; and (iv) the valley-free betweenness centrality is a good metric for selecting measurement ASes.

Future work includes using our proposals to implement a system capable of continuously monitoring TD in the Internet. That could be for instance a crowdsourcing system, in which participating users report measurements and rely on the system to monitor whether they are being victims of TD; those users should also allow their devices to be used as measurement points for other users. Another direction is the development of a hybrid version of our solution using *traceroute*-like techniques: if the exact path between ASes can be obtained, it may not be necessary to consider all possible paths. Furthermore, our proposals consider only TD based on application. Detecting and locating TD based on the origin/destination is still an under-explored topic. Evaluating our proposals on scenarios where multiple ASes may be discriminatory is also left for future work. Finally, another research direction is to design a system that, after locating which AS is discriminating traffic, deviates traffic through a path known to be fully neutral, circumventing the discriminatory AS.

CRediT authorship contribution statement

Thiago Garrett: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft, Writing – review & editing. **Luis C. E. Bona:** Conceptualization, Resources, Writing – review & editing, Supervision. **Elias P. Duarte Jr.:** Conceptualization, Resources, Writing – review & editing, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

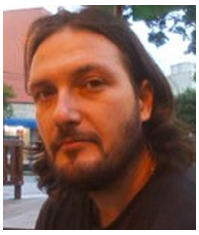
This work was partially supported by grants 308959/2020-5 and 307886/2019-0 from the Brazilian Research Council (CNPq), as well as the Brazilian Ministry of Education (CAPES), finance code 001.

References

[1] T. Wu, A proposal for network neutrality, 2002, URL http://www.timwu.org/OriginalNNProposal.pdf, Accessed on July 12th, 2021.
[2] S. Dustdar, E.P. Duarte, Network neutrality and its impact on innovation, IEEE Internet Comput. 22 (6) (2018) 5–7, http://dx.doi.org/10.1109/MIC.2018.2877838.
[3] T. Berners-Lee, Long live the web, Sci. Am. 303 (6) (2010) 80–85.
[4] B. van Schewick, D. Farber, Point/counterpoint: Network neutrality nuances, Commun. ACM 52 (2) (2009) 31–37.
[5] H. Schulzrinne, Network neutrality is about money, not packets, IEEE Internet Comput. 22 (6) (2018) http://dx.doi.org/10.1109/MIC.2018.2877837.
[6] J.M. Bauer, G. Knieps, Complementary innovation and network neutrality, Telecommun. Policy 42 (2) (2018) 172–183, http://dx.doi.org/10.1016/j.telpol.2017.11.006.
[7] H. Habibi Gharakheili, A. Vishwanath, V. Sivaraman, Perspectives on net neutrality and internet fast-lanes, SIGCOMM Comput. Commun. Rev. 46 (1) (2016) 64–69.
[8] T. Garrett, L.E. Setenareski, L.M. Peres, L.C.E. Bona, E.P. Duarte, Monitoring network neutrality: A survey on traffic differentiation detection, IEEE Commun. Surv. Tuts. 20 (3) (2018) http://dx.doi.org/10.1109/COMST.2018.2812641.
[9] Y. Zhang, Z.M. Mao, M. Zhang, Detecting traffic differentiation in backbone ISPs with NetPolice, in: ACM SIGCOMM IMC, 2009.
[10] R. Ravaioli, G. Urvoy-Keller, C. Barakat, Towards a general solution for detecting traffic differentiation at the internet access, in: International Teletraffic Congress (ITC), 2015, http://dx.doi.org/10.1109/ITC.2015.8.
[11] E. Gregori, V. Luconi, A. Vecchio, Studying forwarding differences in european mobile broadband with a net neutrality perspective, in: European Wireless Conference, 2018.
[12] Z. Zhang, O. Mara, K. Argyraki, Network neutrality inference, ACM SIGCOMM Comput. Commun. Rev. 44 (4) (2014).
[13] S. Cho, R. Nithyanand, A. Razaghpanah, P. Gill, A churn for the better: Localizing censorship using network-level path churn and network tomography, in: ACM CoNEXT, 2017, http://dx.doi.org/10.1145/3143361.3143386.
[14] T. Garrett, L.C.E. Bona, E.P. Duarte Jr., Exploiting AS-level routing properties to locate traffic differentiation in the internet, in: IEEE Symposium on Computers and Communications (ISCC), 2020.
[15] P. Gill, M. Schapira, S. Goldberg, A survey of interdomain routing policies, ACM SIGCOMM Comput. Commun. Rev. 44 (1) (2013) http://dx.doi.org/10.1145/2567561.2567566.
[16] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, M. Bowman, PlanetLab: An overlay testbed for broad-coverage services, ACM SIGCOMM Comput. Commun. Rev. 33 (3) (2003).
[17] R. Beverly, S. Bauer, A. Berger, The internet is not a big truck: Toward quantifying network neutrality, in: Internatinoal Conference on Passive and Active Network Measurement (PAM), Springer, 2007, pp. 135–144.
[18] M.B. Tariq, M. Motiwala, N. Feamster, NANO: Network access neutrality observatory, in: 7th ACM Workshop on Hot Topics in Networks (Hotnets-VII), 2008.
[19] G. Lu, Y. Chen, S. Birrer, F.E. Bustamante, X. Li, POPI: A user-level tool for inferring router packet forwarding priority, IEEE/ACM Trans. Netw. 18 (1) (2010) 1–14.
[20] P. Kanuparthy, C. Dovrolis, DiffProbe: Detecting ISP service discrimination, in: IEEE INFOCOM, 2010, pp. 1–9.
[21] M. Dischinger, M. Marcon, S. Guha, K.P. Gummadi, R. Mahajan, S. Saroiu, Glasnost: Enabling end users to detect traffic differentiation, in: USENIX Conference on Networked Systems Design and Implementation, USENIX Association, 2010, p. 27.
[22] U. Weinsberg, A. Soule, L. Massoulie, Inferring traffic shaping and policy parameters using end host measurements, in: IEEE INFOCOM, 2011, pp. 151–155.
[23] A. Molavi Kakhki, A. Razaghpanah, A. Li, H. Koo, R. Golani, D. Choffnes, P. Gill, A. Mislove, Identifying traffic differentiation in mobile networks, in: Internet Measurement Conference, ACM, 2015, pp. 239–251.
[24] F. Li, A.A. Niaki, D. Choffnes, P. Gill, A. Mislove, A large-scale analysis of deployed traffic differentiation practices, in: ACM Special Interest Group on Data Communication, in: SIGCOMM '19, ACM, 2019, pp. 130–144, http://dx.doi.org/10.1145/3341302.3342092.
[25] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, R. Teixeira, Avoiding traceroute anomalies with Paris traceroute, in: ACM SIGCOMM IMC, 2006, http://dx.doi.org/10.1145/1177080.1177100.

[26] M. Luckie, B. Huffaker, A. Dhamdhere, V. Giotsas, k. claffy, AS relationships, customer cones, and validation, in: ACM SIGCOMM IMC, 2013, http://dx.doi.org/10.1145/2504730.2504735.
[27] T. Garrett, S. Dustdar, L.C.E. Bona, E.P. Duarte Jr., Ensuring network neutrality for future distributed systems, in: International Conference on Distributed Computing Systems (ICDCS), 2017, pp. 1780–1786.
[28] CAIDA, CAIDA AS rank, 2019, Accessed on July 12th, 2021, https://asrank.caida.org/.
[29] M.E. Tozal, Enumerating single destination, policy-preferred paths in AS-level internet topology maps, in: IEEE Sarnoff Symposium, 2016, http://dx.doi.org/10.1109/SARNOF.2016.7846759.
[30] Z. Frias, J.P. Martínez, 5G networks: Will technology and policy collide? Telecommun. Policy 42 (8) (2018) 612–621, http://dx.doi.org/10.1016/j.telpol.2017.06.003.
[31] V. Giotsas, S. Zhou, Valley-free violation in internet routing – analysis based on bgp community data, in: IEEE International Conference on Communications (ICC), 2012, http://dx.doi.org/10.1109/ICC.2012.6363987.
[32] E. Gregori, V. Luconi, A. Vecchio, NeutMon: Studying neutrality in European mobile networks, in: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2018, pp. 523–528, http://dx.doi.org/10.1109/INFCOMW.2018.8407022.
[33] V. Pejović, Towards a holistic net neutrality violation detection system: A case study of Slovenia, J. Netw. Syst. Manage. 28 (2020) 1453–1481.
[34] R. Rizzi, G. Sacomoto, M.-F. Sagot, Efficiently listing bounded length st-paths, in: K. Jan, M. Miller, D. Froncek (Eds.), Combinatorial Algorithms, Springer International Publishing, Cham, 2015, pp. 318–329.
[35] U. Brandes, A faster algorithm for betweenness centrality, J. Math. Sociol. 25 (2) (2001) http://dx.doi.org/10.1080/0022250X.2001.9990249.
[36] CAIDA, Routeviews prefix to AS mappings dataset for IPv4 and IPv6, 2018, Accessed on July 12th, 2021, http://www.caida.org/data/routing/routeviews-prefix2as.xml.
[37] Peeringdb, 2021, Accessed on July 12th, 2021, http://www.peeringdb.com.
[38] A. Lodhi, N. Larson, A. Dhamdhere, C. Dovrolis, k. claffy, Using peeringdb to understand the peering ecosystem, ACM SIGCOMM Comput. Commun. Rev. 44 (2) (2014) http://dx.doi.org/10.1145/2602204.2602208.

Thiago Garrett is a Postdoctoral Research Fellow at the University of Oslo, Norway, where he is a member of the Networks and Distributed Systems research group. He received the degrees B.Sc., M.Sc., and Ph.D. in Computer Science from the Federal University of Paraná, Curitiba, Brazil, in 2008, 2011, and 2019, respectively. His research interests include Computer Networks, Distributed Systems, Internet of Things, and Blockchain technologies.



Luis C. E. Bona is an Associate Professor at Federal University of Paraná, Curitiba, Brazil, where he is member of the Computer Networks and Distributed Systems Lab (LaRSis). He obtained a Ph.D. degree in Electrical Engineering at Federal University of Technology - Paraná, 2006, and carried out his post-doctoral studies at the Barcelona Supercomputing Center (BSC), 2013. His research interests include Operating Systems, Computer Networks and Distributed Systems. He acted as coordinator of several research, technological and development projects, both national and international. He also served as chair of the Department of Computer Science of Federal University of Paraná from 2008 to 2012.

Elias P. Duarte Jr is a Full Professor at Federal University of Parana, Curitiba, Brazil, where he is the leader of the Computer Networks and Distributed Systems Lab (LaRSis). His research interests include Computer Networks and Distributed Systems, their Dependability, Management, and Algorithms. He has published more than 250 peer-reviewer papers and has supervised more than 130 students both on the graduate and undergraduate levels. Prof. Duarte is currently Associate Editor of the Computing (Springer) journal and IEEE Transactions on Dependable and Secure Computing, and has served as chair of more 25 conferences and workshops in his fields of interest. He received a Ph.D. in Computer Science from Tokyo Institute of Technology, Japan, 1997, M.Sc. in Telecommunications from the Polytechnical University of Madrid, Spain, 1991, and both B.Sc. and M.Sc. degrees in Computer Science from Federal University of Minas Gerais, Brazil, 1987 and 1991, respectively. He chaired the Special Interest Group on Fault Tolerant Computing of the Brazilian Computing Society (2005–2007); the Graduate Program in Computer Science of UFPR (2006–2008); and the Brazilian National Laboratory on Computer Networks (2012–2016). He is a member of the Brazilian Computing Society and a Senior Member of the IEEE.