MAP-IT: Multipass Accurate Passive Inferences from Traceroute

Alexander Marder
University of Pennsylvania
amarder@seas.upenn.edu

Jonathan M. Smith
University of Pennsylvania
jms@seas.upenn.edu

ABSTRACT

Mapping the Internet at scale is increasingly important to network security, failure diagnosis, and performance analysis, yet remains challenging. Accurately determining the interface addresses used for inter-AS links from traceroute traces can be hard because these interfaces are often assigned addresses from neighboring ASes. Identifying these interfaces can benefit Internet researchers and network diagnosticians by providing accurate IP-to-AS mappings where such mapping is most difficult – at AS boundaries.

We describe a new algorithm, Multipass Accurate Passive Inferences from Traceroute (MAP-IT), for inferring the exact interface addresses used for point-to-point inter-AS links, as well as the specific ASes involved. MAP-IT combines evidence of an AS switch from distinct traceroute traces; using traceroute data makes it portable across IP networks. Each pass leverages prior inferences to refine existing inferences and to discover additional inter-AS link interfaces.

We test MAP-IT with interface-level ground truth information from Internet2, achieving 100% precision. Using approximate ground truth from Level 3 and TeliaSonera yields 95.0% precision. These results suggest that MAP-IT is sufficiently reliable for network diagnostics.

1. INTRODUCTION

Some points in the global Internet are particularly important for reliability. For example at AS boundaries, disruptions (such as via DDoS) are particularly effective. Identifying the interfaces at AS boundaries [32], where they are often assigned addresses belonging to neighboring ASes [35] has proven challenging. Two distinct problems conspire to make this especially difficult, which are coarse-grained interface-to-AS mappings using IP prefixes [40], and incomplete and inaccurate interface-to-router assignments [21]. Reliably identifying these inter-AS link interface addresses may seem like a narrow problem, but is central to many problem domains, as it underlies the accuracy of those studies or solutions. Examples include:

- studies that rely on identifying inter-AS link interfaces, such as measuring congestion on peering links [32] and mapping interfaces to facilities [19];
- more precisely identifying the ASes traversed on a traceroute path, with implications for AS-connectivity research and network diagnosis [35]; and
- studying the security implications of network topologies against flooding and DDoS attacks [27].

Unfortunately, existing techniques are too inaccurate to provide this type of information.

1.1 Difficulties at the Boundary

There are two reasons that inferring inter-AS link interfaces from a single trace is difficult. First, a link between two ASes is assigned addresses from one address space. As a result, the address at which the AS address space changes from one AS to another in a trace is not always the address used for the interconnection between the two. In fact, when a path has a single hop in an AS, an address from its address space may not appear in the trace.

Second, traceroute artifacts may lead to errors. Load balancing has the potential to render the inter-AS link interface address invisible in some traces, and routers might respond with off-path third party addresses, which both hides the address used for the interconnection and can cause an extra AS to appear in the trace.

Clearly, it is unwise to draw inferences about inter-AS links from a single trace. Additional information, i.e., traces that expose additional interfaces either before or after the same interface, is needed to draw accurate inferences.

To address these challenges, we provide a technique for inferring the interfaces used for inter-AS links. From an interface-level graph we derive from a set of traceroute traces, we identify the interfaces in the graph where

IMC 2016 November 14-16, 2016, Santa Monica, CA, USA
© 2016 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-4526-2/16/11.
DOI: http://dx.doi.org/10.1145/2987443.2987468

an AS switch occurs. We then improve these inferences by making multiple passes through the graph. This approach effectively handles the challenges posed by naming point-to-point interfaces from `/30` or `/31` prefixes, minimizes the impact of third party addresses and load balancing, and corrects for mistaken inferences due to low visibility.

1.2 Contributions

In this paper, we present MAP-IT, which is an algorithm that identifies inter-AS link interfaces, and the ASes connected by the link. We evaluate our algorithm using data available to the research community, ensuring that our results can be reproduced. In addressing this challenge we:

- describe a novel, robust, and highly precise, multipass algorithm for inferring interfaces used for inter-AS links from traceroutes;
- verify our algorithm using ground truth from Internet2, a Tier 2 regional provider, achieving a precision of 100% and a recall of 96.9%;
- confirm those results using approximate ground truth derived from DNS hostnames for interfaces in ISPs Level 3 and TeliaSonera; and
- compare MAP-IT to existing approaches for identifying inter-AS links interfaces, demonstrating better accuracy.

Our verification against ground truth data confirms the ability of our algorithm to find inter-AS link interfaces (recall) when possible in the traceroute dataset, and the correctness of those inferences (precision).

2. RELATED WORK

Extracting Information From DNS: DNS hostnames can provide useful information for interpreting traceroute results. Previous analysis [14, 23] shows that they do not follow universal tagging conventions, making automation of extracting pertinent information harder. Furthermore, hostnames are not regularly updated, leading to stale information [41]; some networks provide no information in their hostnames; and many interfaces lack associated DNS hostnames. These problems conspire to make using hostnames as a standalone technique infeasible, and make even using them in combination with other methods difficult.

Router-Level Graphs: Significant efforts were made to leverage the router-level graphs generated by alias resolution techniques [13, 20, 28, 30, 38]. Unfortunately, even accurately inferred routers are difficult to map to ASes [15], confounding their ability to accurately infer the interface addresses used for specific inter-AS links. Huffaker, et al. [21] use MIDAR [30] and `kapar` [29] to group interface addresses into routers, and propose several heuristics for router-to-AS assignments, achieving 71% router-to-AS mapping accuracy. Giotsas, et al. [19] use a similar technique, without `kapar`, to infer inter-AS links. These links are input to their Constrained Facility Search algorithm, which iteratively constrains the possible interconnection facilities for inferred peerings. They do not evaluate the accuracy of their inferred inter-AS links.

A highly accurate router-level graph would mitigate many of the challenges traceroute presents [35]. Currently, none of the router-level graphs are accurate enough for use in inferring inter-AS links. In this paper, we adopt a different approach that avoids alias resolution entirely. Namely, we use clues at the interface-level that an interface is used for an inter-AS link.

IP-to-AS Mapping: Others have recognized the problems of prefix-based IP-to-AS mappings for traceroute interface addresses. Mao, et al. [35] classify the various causes of AS-level path mismatches between BGP and traceroute, and later propose a technique to reassign /24 prefixes that might be mis-mapped, focusing primarily on missing AS hops when compared to BGP announcements, caused by MOAS prefixes, IXP prefixes, and sibling ASes [34]. A subsequent study [40] refines this method by reassigning interfaces at the address granularity, but focuses on altering traceroute derived AS-level paths to better reflect BGP derived AS-level paths, which may be inaccurate. Zhang, et al., [42, 43] propose a framework for quantifying discrepancies between traceroute-derived AS paths and BGP AS paths, and find that 60% of mismatches are due to ASes assigning interfaces from a prefix announced by a neighbor.

AS-Level Links/AS Connectivity: Chen, et al. [16] propose using traceroute to complement the AS-level links derived from BGP route announcements, from which AS connectivity is typically inferred. They also provide heuristics for converting traceroute IP-paths to AS-level paths, which enables them to avoid false positives when compared to ground truth from a Tier 1 network.

While deriving the existence of a link between ASes is related to our work, these heuristics cannot be used to identify the IP addresses used on the inter-AS links, nor to identify all IP-level links that appear in traceroute traces between two networks. Our algorithm addresses the problem of identifying legitimate AS-level links as well as the IP addresses used on those links.

Third Party Addresses: Third party addresses may appear in traceroute traces when the ingress interface which receives a probe and the egress interface used for the ICMP response are different. Hyun, et al. [25] attempt to quantify the prevalence of third party addresses announced by off-path ASes, concluding that third party addresses are primarily caused by multihomed ASes, and do not significantly distort the AS-level paths derived from traceroute. Employing the IP prespecified timestamp option, Marchetta, et al. [36] measure the impact of third party addresses on inferred AS-links derived from traceroute, determining that 17% of the AS-links are mistaken inferences caused by third party addresses. They argue that third party ASes are much more prevalent than found by Hyun, et al., and that they are not limited to the edges of the Internet.

```
2 isc-ist.seas.upenn.edu (158.130.0.250) [AS55]
3 vag-brdr.i2trcps-ashb.router.upenn.edu (128.91.238.222) [AS55]
4 72.14.217.16 (72.14.217.16) [AS15169]
```

Figure 1: Abridged traceroute output showing hops 2 through 4, with DNS hostnames (when possible), and the origin AS derived from BGP prefix announcements.

A follow up study by Luckie and Claffy [31] questions these results, arguing that the IP prespecified timestamp is non-standard and thus cannot be relied upon for accurate measurements. They demonstrate that more than half of the addresses inferred to be off-path by Marchetta, et al. are in fact on-path addresses.

These studies show that the impact of third party addresses is not completely understood, and further, that a reliable active or passive approach for identifying third party ASes in traceroute traces has not yet been found. While one could try to identify third party addresses initially, we minimize their impact on our results.

Inter-AS Link Interfaces: Luckie, et al. [32] discuss the challenges of measuring congestion on peering links, but did not present a method for identifying inter-AS link interfaces. In later work, Luckie, et al. propose `bdrmap` [33] to tackle the problem of inferring inter-AS link interface addresses between a network with at least one traceroute monitor and *directly* connected networks, with 96.3%-98.9% precision. MAP-IT, unlike `bdrmap`, tries to identify inter-AS link interfaces between *all* connected ASes seen in traceroute results, not just for directly connected networks. Of the three networks we verify against, only one has a monitor that was used to run the traceroutes for our experiments.

3. IMPROVING INFERENCES

The two interfaces connected by a layer 3 point-to-point link are assigned addresses from the same `/30` or `/31` prefix [37]. When applying BGP-based IP2AS mappings to interfaces used on AS interconnection links, one interface will map to the incorrect AS, since the prefix is allocated to only one AS. Using Fig 1 as an example, the interfaces at hops 2 and 3 map to AS55 (University of Pennsylvania), and the fourth interface maps to AS15169 (Google). These mappings, derived from BGP prefix announcements, might lead to the (mistaken) inference that AS55 connects directly to AS15169. In fact, AS55 indirectly connects to AS15169 via AS11164 (Internet2 TR-CPS), which the DNS hostnames reveal. Complicating matters, here a transit link is assigned a prefix from the customer's address space (AS55), violating the convention that the link prefix is typically assigned from the provider's address space [21, 32].

As Fig 1 shows, inferring inter-AS links from a single trace is unreliable. Rather, combining evidence from separate traces is necessary to form a more trustworthy interface graph.

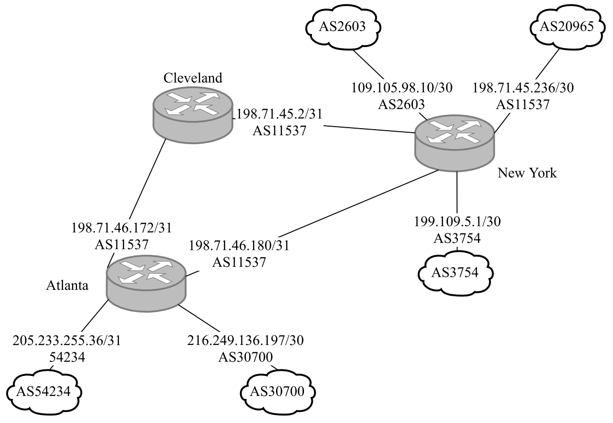


Figure 2: Sample of traces combined with ground truth from Internet2. Packets move down and to the left.

3.1 A New Approach

This section explains the MAP-IT approach. § 1 outlined the problem and our goals, and here we describe design choices we have made, before presenting MAP-IT details. The most significant is avoiding additional infrastructure, such as the probes used by Luckie, et al. [33], and instead focusing on algorithmic advances. MAP-IT consequently is usable in today's environments, and provides a road map to further improve accuracy when such infrastructure is available. Using existing data not specifically collected for MAP-IT presumes that the information we need can be extracted via processing. Traceroute data collections provide information about paths through the Internet, and identification of AS boundaries demands both transforming the data from a list of probes into a graph, and then using the same data to identify points in the graph with the desired properties. As with any data-driven approach, sufficient data of sufficient quality are necessary, but the basic approach is to build a graph, identify likely AS boundaries, and then extract additional information from the traceroute files. In MAP-IT, this extraction process is a central contribution, as we discovered that additional passes through the data could be used to refine the tentative identifications by increasing or decreasing the likelihood that the graph node is a boundary. As we will show when evaluating MAP-IT, this refinement process is robust enough on real data that AS-boundary identification accuracy similar to Luckie, et al. was achieved with no additional infrastructure.

Fig 2 illustrates the outputs from several traces through AS11537 (Internet2). Ingress interface addresses are shown at each router, which are what traceroute generally reports, as well as the AS address space to which they belong[1]. No single trace contains all of the ad-

[1] We say that an interface address is *in*, *from*, or *belongs to* an AS if its longest matching prefix is originally announced by that AS.

dresses displayed, but combining them helps create a clearer picture of the network.

As expected, the **intra**-AS link interface addresses – those used to connect the core routers – are from AS11537. Conversely, the **inter**-AS link addresses are either assigned by AS11537 or the connected AS, illustrating the primary difficulty in inferring the specific interface addresses used for the AS interconnections. A single trace does not provide sufficient data to determine which AS's address space is used for the inter-AS link interfaces, but the aggregate view provides clues from which we can derive more accurate inferences.

For the interface address `109.105.98.10`, the unique addresses that appear after it (forward neighbors set, $\mathbb{N}_F$) in the traces are primarily from AS11537. Although the address is assigned from AS2603 (NORDUnet), its $\mathbb{N}_F$ implies that it resides on a router in AS11537. The rationale behind this inference is that if `109.105.98.10` was an internal interface, most of the forward neighbors would not belong to a single, different AS. The next hop following a border router is typically another router in the same AS that responds with an interface address from its own address space, which in this case are internal addresses on the Cleveland and Atlanta routers. As the link addresses will come from one of the two interconnected ASes, and we found that `109.105.98.10` is used on an AS11537 router, we can infer that it is used to connect to a router in AS2603. This reasoning also applies to the unique neighboring addresses that appear before (backwards neighbors set, $\mathbb{N}_B$) an inter-AS link interface, as demonstrated by the interfaces `205.233.255.36` and `216.249.136.196`.

Such clues will only appear before or after an inter-AS link interface. Internal interfaces, such as `198.71.46.180`, are not expected to have a single AS dominate their neighbors sets ($\mathbb{N}$s), other than the AS which controls their address space. Accordingly, both its $\mathbb{N}_F$ and $\mathbb{N}_B$ contain two addresses from two different networks. Although its $\mathbb{N}$s indicate that the interface is used internally, we do not attempt to identify internal interfaces, noting this only to point out the differences between inter-AS links and intra-AS links.

Returning to inter-AS links, for point-to-point links we may only see inter-AS link clues in one direction, i.e., in either $\mathbb{N}_F$ or $\mathbb{N}_B$. These interface addresses are uniquely used to connect two routers[2]. While the $\mathbb{N}_F$ for `109.105.98.10` will contain interfaces on the far side of the links that connect routers to the New York router, in this case `198.71.45.2`, `198.71.46.180` and `199.109.5.1`, its $\mathbb{N}_B$ should contain interface addresses seen on the single router preceding it in AS2603, causing its $\mathbb{N}_B$ to resemble those of internal interfaces.

Importantly, the $\mathbb{N}_F$ and $\mathbb{N}_B$ for the same interface are expected to be disjoint regardless of the vantage point from which probes are sent; but different vantage points can reveal additional interfaces before and after an interface. This occurs because traceroute is intended to report the ingress interface addresses on which the routers receive the probes, so for point-to-point links, the backward neighbors and forward neighbors should be disjoint sets of interface addresses[3]; otherwise, there would be a forwarding loop between two of the routers.

Further, evidence of an inter-AS link not only allows us to infer the use of the interface address we are looking at, but also the use of the interface on the other side of the link. For point-to-point addresses, the other side is assigned an address from the other host address in its `/30` or `/31` prefix[4]. Since the other side is used on the same inter-AS link, it is necessarily used to connect the same two ASes. In Fig 2, the other side of `109.105.98.10` is `109.105.98.9`, and is used on the same link connecting AS11537 to AS2603. The former address resides on a router in AS11537, while the latter's router is in AS2603.

Finally, nothing can be inferred from the $\mathbb{N}_B$ for the inter-AS link interface `199.109.5.1`, since no single AS appears more than all others. However, when we update the IP2AS mappings based on previous inferences in § 4.4.1, we will be able to determine that it is used for the link between AS11537 and AS3754 (NYSERNet) on a subsequent pass through the interfaces.

3.2 Interface Halves

We attempt to infer the use of an interface address based on each of its $\mathbb{N}$s independently, because only one direction is expected to indicate its use on an inter-AS link. For this reason we split each interface into interface halves (IHs), where each half is the interface in either the forward or backward direction. Forward halves include only the forward neighbors for an interface, while backward halves can see only the backward neighbors. The other side of an IH is the other side of the original interface, but looking in the opposite direction. As an example, the other side of the backward half of the interface `198.71.46.180/31` is the forward half of the interface `198.71.46.181/31`. This becomes important when we update the IP2AS mappings in § 4.4.2. When refering to an interface i, we use i_f and i_b to refer to its forward and backward halves.

Using Fig 3 as an example, we split the interface `198.71.46.180` into the forward half `198.71.46.180`$_f$ and

[2] This is true for globally routable addresses, but not for private/shared addresses [17], which can be reused by many ASes. For this reason we do not make inferences on private/shared interface addresses.

[3] Traceroute artifacts, such as per-packet load balancing and routers responding with their outgoing interface, can cause an interface to be in both $\mathbb{N}_F$ and $\mathbb{N}_B$. In our experiments, 0.3% had an interface in both $\mathbb{N}$s.

[4] In a `/30` prefix, only the middle two addresses can be used as host addresses, since for all prefixes shorter than a `/31`, the first and last addresses are reserved. To preserve the IPv4 address space, RFC 3021 [37] permits both addresses in a `/31` prefix to be host addresses in a point-to-point link.

Paths

1:	109.105.98.10	**198.71.46.180**	205.233.255.36
2:	109.105.98.10	**198.71.46.180**	216.249.136.197
3:	198.71.45.236	**198.71.46.180**	*
4:	109.105.98.10	**198.71.46.180**	199.109.5.1

Neighbors Sets

198.71.46.180$_f$: $\{$205.233.255.36$_b$, 216.249.136.197$_b$, 199.109.5.1$_b\}$
198.71.46.180$_b$: $\{$109.105.98.10$_f$, 198.71.45.236$_f\}$

Figure 3: Original path segments and the resulting interface halves for the interface address 198.71.46.180.

the backward half 198.71.46.180$_b$. Two things should be noted. First, information from incomplete paths is included when creating the $\mathbb{N}$s, as evidenced by the inclusion of 198.71.45.236 in $\mathbb{N}_B$. Second, the $\mathbb{N}$s only include unique addresses, and the number of traces in which an address appears is not reflected in $\mathbb{N}$. For this reason, 109.105.98.10$_f$ is included in the $\mathbb{N}_B$ once.

3.3 Discussion

This approach is intended to find the IP connections between networks, which can be physical, such as a cross-connect in a co-location facility, or virtual, where two routers are connected by a VLAN. Common examples of a virtual point-to-point connection are virtual private interconnections established across a peering fabric at an Internet Exchange Point (IXP), and data centers which connect networks to transit providers over a switched network. These switching fabrics are invisible to traceroute, which only reports the IP-layer interfaces, so although the two networks are not directly connected to each other, the VLANs appear like physical point-to-point inter-AS link interfaces. The impact is that this technique infers the logical AS interconnections, but not the switched network that enables those connections.

MAP-IT relies on $\mathbb{N}$s with at least two addresses from which to draw inferences, except for stub ASes (see § 4.8). This helps avoid incorrect inferences due to traceroute artifacts, described in greater detail in the next section. Where only one address from the other address space is seen before or after an inter-AS link between ISP ASes, an inter-AS link cannot be inferred.

The need to see interfaces in the address space of the connected AS causes two problems. First, when an AS uses interface addresses from its transit provider, we are unable to infer an inter-AS link because the address space does not change. Second, some ASes disable traceroute replies on their border routers, which prevents us from making inferences.

4. MULTI-PASS ALGORITHM

We now present MAP-IT, an algorithm for inferring inter-AS link interfaces. We presume traces and an IP2AS mapping tool based on BGP prefix announcements are available.

Alg 1 sketches MAP-IT. We remove spurious or anomalous traceroute results (line 1), and use the remaining sanitized traces to create $\mathbb{N}_F$ and $\mathbb{N}_B$ for each interface (line 2). Subsequently, we make as many inferences as possible (line 4), updating our IP2AS mappings as we go; prune disproved inferences (line 5); and repeat until the algorithm converges. Finally, we employ a heuristic to infer links involving low visibility stub ASes or stub ASes which employ NATs (line 7).

Algorithm 1 MAP-IT

```
1: Sanitize traces to remove artifacts (§ 4.1)
2: Create $\mathbb{N}_F$ and $\mathbb{N}_B$ (§ 4.3)
3: repeat
4:     Add inter-AS link inferences (§ 4.4)
5:     Remove questionable inferences (§ 4.5)
6: until there are no changes left to make
7: Infer links to low visibility and NAT stubs (§ 4.8)
```

4.1 Discard and Sanitize Traces

Router misconfigurations, load balancing, and transient routing changes can cause traceroute artifacts. We take two steps to minimize their pollution of the $\mathbb{N}$s.

First, we try to remove false adjacencies caused by buggy routers that forward packets with TTL=1, instead of sending an ICMP reply [26], as these may make interfaces appear to be on adjacent routers, when there is an unseen router. We remove all hops with quoted TTL=0, but retain the rest of the trace.

After sanitizing a trace, we attempt to identify if load balancing or a transient routing change occurred during the trace. While Paris traceroute [39] mitigates the effects of most load balancing techniques, per-packet load balancing and transient routing changes can cause mistakes in the $\mathbb{N}_F$ and $\mathbb{N}_B$ by making interfaces on disconnected routers appear adjacent in a trace. We discard traces that have an interface cycle[5]. For the traces we used, we remove 2.7% of the traces and retain 89.1% of the distinct addresses seen in the dataset.

What remain are cleaner but not perfect traces. This is acceptable as MAP-IT is robust to moderate misinformation.

4.2 Determine Interface Other Sides

As noted, point-to-point links can be assigned addresses from either a /30 or /31 prefix, so we use a heuristic to infer the other side for each interface. From a traceroute dataset, we extract all addresses seen in any trace, including discarded traces. All non-host addresses in a /30 prefix are assigned an other side from their /31 prefix. For the remaining valid host address, we check to see if a different address appeared in our dataset that would be a reserved address in its /30 prefix. If so, we assign it an other side from its /31 prefix, otherwise we assume it is from a /30 prefix. In total, this heuristic identifies 40.4% of the interfaces as being addressed from a /31 prefix.

[5] A cycle [39] is where the same address appears twice, separated by at least one other address.

Algorithm 2 Direct Interfaces

```
Require: $f$                                                    ▷ $0 \le f \le 1$
for each IH, $h$, w/o a direct inference do
    Find $AS_N$ which appears more than any other AS in $h$'s $\mathbb{N}$ using previous IP2AS
    if COUNT($AS_N$) $\ge$ COUNT($neighbors$) $\times f$ then
        if previous IP2AS($h$) $\ne AS_N$ then
            Mark a direct inference for $h$
            Update current IP2AS($h$) $\leftarrow AS_N$
```

4.3 Extract Neighbors Sets

Using the cleaned traces, we create $\mathbb{N}_F$ and $\mathbb{N}_B$ described in § 3. The $\mathbb{N}$s for an interface includes all addresses seen exactly one hop before it ($\mathbb{N}_B$) or after it ($\mathbb{N}_F$) across all traces, excluding `null` hops and private/shared addresses. We do not include private/shared addresses in the $\mathbb{N}$s, because they are not globally routable or unique, and should not appear in traceroutes.

Of the 4,752,201 interface addresses seen adjacent to at least one other address in our dataset, 449,602 had $\mathbb{N}_F$ with more than one address, and 1,139,087 had $\mathbb{N}_B$ with more than one address. For a direct inference to be made in § 4.4.1 on a specific interface, either its $\mathbb{N}_F$ or $\mathbb{N}_B$ must contain at least 2 addresses.

4.4 Adding Inferences

This section describes the add step, the main process for inferring inter-AS links. An interface is determined to be used for an inter-AS link in 4 steps:

1. Use the $\mathbb{N}$s and the current IP2AS mappings to make direct inferences on the IHs (§ 4.4.1);
2. Update the mappings for the other side of each direct inference (§ 4.4.2);
3. Resolve contradictions where inferences are made in the forward and backward directions for the same interface (§ 4.4.3);
4. Resolve inverse inferences[6] made on adjacent IHs, retaining one inference and discarding the other (§ 4.4.4).

This continues until no additional inferences can be made.

4.4.1 *Direct Inferences*

Alg 2 shows the first step, which greedily tries to make as many inferences as possible on the set of IHs with more than one neighbor in their $\mathbb{N}$s, based on the current IP2AS mappings of their neighbors. As we visit each IH, we map each IH in its $\mathbb{N}$ to an AS using the IP2AS mapping tool. For the first pass through the IHs, this is equivalent to the IP2AS mappings derived from BGP announcements. If any single AS appears more than all other ASes, and that AS is not the same as the IH's AS, then a direct inference is made.

When counting the ASes in a $\mathbb{N}$ we treat all sibling ASes like they are the same AS. We do not distinguish between siblings as they are controlled by the same organization, and the siblings do not necessarily assign addresses according to the distinctions between the ASes. If an inference is made for a $\mathbb{N}$ with multiple siblings from the same organization, we update the IP2AS mapping with the sibling AS that appears most frequently in $\mathbb{N}$.

The parameter $0 \le f \le 1$ can be used to further restrict the inferences. After determining the most common AS in $\mathbb{N}$, at least $f \times |\mathbb{N}|$ of the addresses in $\mathbb{N}$ must map to that AS for an inference to be made. As an example, for $f = 0.5$, at least half of the addresses in $\mathbb{N}$ must map to the most common AS, or the inference is discarded. We evaluate the impact of f in § 5.3, showing that the fraction of correct inferences generally improves as f increases, at the expense of the number of inter-AS links identified.

When an inference is made, two things happen. First, we record the inference and the AS connected via the link as a direct inference. This prevents the algorithm from making a different direct inference on the IH, unless the inference is removed in § 4.5. This is important to prevent update cycles on the same IH. Second, we update the IP2AS mapping for the IH to the connected AS. The update is applied whether it is a forward or backward inference, as they enable future updates.

Returning to Fig 2, during the initial pass through the IHs, no inference can be made for `199.109.5.1`$_b$ because no AS appears more than all other ASes in its $\mathbb{N}_B$. But, after we make a direct inference on the IH `109.105.98.10`$_f$, we update its IP2AS mapping from AS2603 to AS11537. Now, on the next pass through the IHs, AS11537 appears more than any other AS in the $\mathbb{N}$ of `199.109.5.1`$_b$, providing the clue needed to infer that it is used for an inter-AS link between AS11537 and AS3754. To ensure determinism, all updates made to the IP2AS mappings are only visible starting with the next iteration through the interface halves, so an update made during the first iteration is only used starting with the second iteration.

An IP2AS update on one half of an interface does not affect the IP2AS mapping for the other half. Updating the IP2AS mappings this way can aid in making inferences in later iterations, and prevent mistaken inferences due to skipping an intermediate AS. In the example, while the inference on `198.71.45.236`$_b$ updates its IP2AS mapping to AS20965, which is only seen by members of its $\mathbb{N}_B$, its forward half does not receive that update. Only updating the backward half enables us to make an inference on `199.109.5.1`$_b$, because while `109.105.98.10`$_f$ is in its $\mathbb{N}$, `109.105.98.10`$_b$ is not.

4.4.2 *Adding Indirect Inferences*

After adding direct inferences, the algorithm updates the IP2AS mappings for the other side of each IH on which an inference was made[7]. For the inference made

[6] If an inference on interface a is made from AS_A to AS_B, then an inverse inference is one made on an interface b from AS_B to AS_A.

[7] This is not true for inferences on known IXP inter-

on the IH 109.105.98.10$_f$, the mapping for 109.105.98.9$_b$ is updated, while for the inference made on 199.109.5.1$_b$ an update is made for 199.109.5.2$_f$. Updating the other sides can help make additional inferences for their neighbors, or lead to the removal of errors, but might create an invalid IP2AS update if the incorrect other side was identified.

The distinction between these inferences and those made directly is that these are indirect inferences linked to their other side. If the associated direct inference is discarded, the indirect inference is also discarded. Additionally, while only a single direct inference can be made on each IH per add step, indirect inferences do not preclude a future direct inference, out of concern that the other side might be incorrect.

4.4.3 Fixing Point-to-Point Contradictions

Point-to-point inter-AS links connect exactly two ASes. Traceroute artifacts, such as routers which respond with the outgoing interface, load balancing, unknown sibling ASes, and unannounced IP addresses, can cause it to appear that some interfaces and/or links connect three or more ASes. It is also possible that the interface is used at an Internet exchange, but the address does not appear in our IXP dataset.

The first type of contradiction, which we refer to as dual inferences, is where two inferences are made on the same interface; i.e. we inferred that both the forward and backward IHs of the same interface are used on inter-AS links. The second contradiction is when direct inferences are made on both an interface and its other side, each involving different connected ASes, which we call divergent other sides. In our experiments, most contradictions involve interfaces and links with IP addresses that do not have mappings in our IP2AS tool. We do not fix these, because assigning IP2AS updates to unannounced IP addresses can enable additional inferences. Our focus here is to resolve the point-to-point contradictions involving interfaces with IP2AS mappings.

Dual Inferences: For dual inferences, where inferences are made in both directions for the same interface, we try to discard one of the inferences, keeping the other. If both inferences involve the same AS, usually caused by per-packet load balancing, sibling ASes, or outgoing interfaces, we retain both as it does not affect the accuracy in terms of the ASes connected by the inter-AS link. Instead, we focus on situations where the inferences involve different ASes, which we expect is caused by routers responding with their outgoing interface, resulting in third party addresses.

Fig 4 shows how a third party address can cause dual inferences. 212.113.9.210 is in AS3356 (Level 3), but appears as a third party address for some traces that travel from AS1299 (TeliaSonera) to AS51159 (Think Systems). This occurs when AS51159 sends the ICMP response back to the traceroute monitor through AS3356, instead of using the interface connected to AS1299 on which the probe packet arrived. While its $\mathbb{N}_F$ indicates that it is used on a router in AS51159 to connect it to AS3356, its $\mathbb{N}_B$ appears to imply that it is used to connect AS3356 to AS1299, due to the fact that addresses from AS1299 appear more than addresses from any other AS. The result is that inferences are made on both the forward half and the backward half of the same interface.

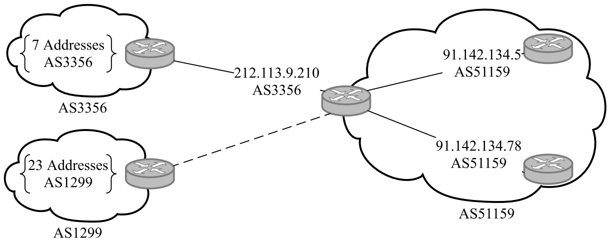


Figure 4: The interface address 212.113.9.210 appears as a third party address for some traces through AS1299, leading to dual inferences.

In this case, and in all dual inferences caused by third party addresses, the forward inference is correct, as the interface that is used to connect the backward AS, AS1299 in this case, is hidden from those traces, causing the backward neighbors from AS1299 to appear erroneously in the $\mathbb{N}_B$. The true interconnection is between AS3356 and AS15119, indicated by the DNS hostname for 212.113.9.210, THINK-SYSTE.edge5.London1.Level3.net. We keep the forward inference and remove the backward inference to resolve the dual inference contradiction.

Resolving dual inferences helps address some of the problems caused by third party addresses, but is unable to account for all of them. Specifically, whenever a forward inference cannot be made for an interface, because its $\mathbb{N}_F$ does not contain enough addresses from the connected AS, there is the possibility that a false backward inference is being made. We expect that false backward inferences are most likely to be made at the borders with stub ASes, as they are often low visibility, which means that the $\mathbb{N}_F$ does not always contain enough addresses. They are also sometimes configured with a single default provider, resulting in ICMP replies traveling through just one of multiple providers.

Another risk is that the forward inference might involve one of its sibling ASes. Although we use a list of known siblings, the list is incomplete. The risk here is that we might remove a legitimate backward inference in favor of the false inference between the siblings.

Divergent Other Sides: The second violation is when different inferences are made on the two endpoints of an inter-AS link. This can occur for the same reasons as dual inferences, with the additional possibility that

faces, because they are often assigned addresses in a multipoint fashion.

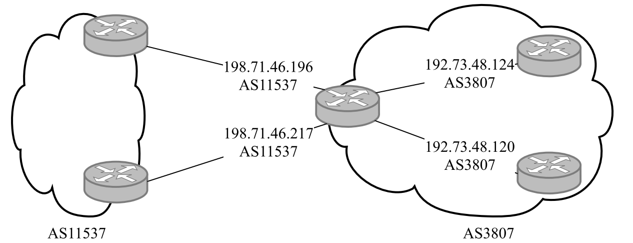


Figure 5: The interface addresses `192.73.48.120` and `192.73.48.124` are inverse inferences between AS11537 and AS3807.

the other side was assigned incorrectly by our heuristic. Since this situation is rare in our experiments, and there is no way to distinguish between the possible causes for a given case, we do not attempt to fix divergent other sides. Instead, we assume the other side is incorrectly identified. Only 90 divergent other sides inferences exist in the final results.

4.4.4 Adjacent Inverse Inferences

Here, we attempt to reduce errors by removing a prominent source of mistaken inferences, "inverse inferences". An inverse inference occurs when a forward inference is made on an interface between its AS and some other AS, and a backward inference is made on a member of its $\mathbb{N}_F$ between the other AS and its AS, as shown in Fig 5. In the figure, both `198.71.46.196`$_f$ and `198.71.46.217`$_f$ are correctly inferred to be used on an inter-AS link between AS11537 and AS3807 (University of Montana). Additionally, `192.73.48.124`$_b$ and `192.73.48.120`$_b$ have $\mathbb{N}$s primarily composed of addresses from AS11537, leading to mistaken inverse inferences. With inverse inferences, typically one of them is wrong and should be discarded. Furthermore, we might encounter inference loops, where the IHs alternately appear to be used for inter-AS links and internal links as the IP2AS mappings are updated.

When confronted with inverse inferences, such as in the example, we keep the inference which is topologically nearer to the traceroute monitors, which is necessarily the forward one. We presume that we would gather more accurate evidence in the $\mathbb{N}_F$ of the nearer interface, making it a more reliable inference. Therefore, we retain the forward inference and discard the backward inferences.

There may be a direct inference made on the other side of the backward IH, such as `192.73.48.21`$_f$ or `192.73.48.25`$_f$ (not shown). In this case, where neither the forward nor the backward IHs are topologically nearer, it is unclear which inference to retain. As we are unable to reliably confirm or discard either inference, MAP-IT classifies both as uncertain inferences. After converging, the algorithm outputs both a list of high confidence inferences, and a much smaller list of uncertain inferences.

The risks associated with removing inverse inferences are twofold. First, our nearness assumption may not always remove the illegitimate inference, causing us to discard a legitimate one instead. Second, traceroute artifacts, such as load balancing, might cause two legitimate inter-AS link interfaces between two networks to appear back-to-back in a single trace, so our attempt to resolve inverse inferences might remove a correct inference.

Algorithm 3 Removing Interfaces

```
repeat
    for each direct inference on $h$ to $AS_N$ do
        if the inference would no longer be made then
            Make the inference indirect
    Discard indirect inferences w/o direct inference
    Remove updates for discarded inferences
until no inferences were discarded
```

4.4.5 Convergence and Determinism

MAP-IT may make many passes through the set of IHs, but since a direct inference on a given half can only be made once, the add step must converge. As inferences are based on updates to the IP2AS mappings prior to the current pass, the inferences are deterministic independent of the order in which the IHs are visited.

4.5 Remove Inferences

Alg 3's pseudocode summarizes the removal of inferences made in the add step. This is necessary to enhance the precision of the algorithm by removing inferences that would no longer be made based on the current IP2AS mappings, and allows the algorithm to revise mappings if necessary.

Similar to the add step, the remove step also makes multiple passes through the set of IHs, only using the IP2AS updates from the previous iteration. As each IH with a direct inference is visited, the algorithm checks if the connected AS still accounts for more than half of its $\mathbb{N}$ based on the current IP2AS mappings. If not, then we initially change the inference from a direct inference to an indirect inference but retain its IP2AS mapping. After each pass through the IHs, all indirect inferences without an associated direct inference are discarded, along with their IP2AS updates.

As with the add step, the remove step is guaranteed to converge because no inferences are made in this step, and once an inference is discarded it remains so at least until the remove step completes. The remove step is deterministic as each pass only uses the updates from the end of the prior pass.

4.6 Overall Convergence

Due to the presence of uncertain inferences, the overall algorithm, which repeats the adding and removing steps, may never reach a point where *no* inferences can be added and *no* inferences can be removed. Instead, it converges to the point where the same inferences are continually added and removed. Therefore, we look for a repeated state at the end of the remove step as the

Algorithm 4 Low Visibility and NAT Heuristic

```
for each IH, $h_f$, w/ a single neighbor $n_b$ do
    if no inference for $h_b$ or $n_b$ & $AS_H \neq AS_N$ then
        if $AS_N$ is a stub AS then
            Mark a direct inference for $h_f$
            Mark an indirect inference for $h'_b$
            Update mappings for $h_f$ and $h'_b$ to $AS_N$
```

stopping criterion, which indicates that no more confident inferences can be made. In our experiments, this occurred after 3 iterations of the main `while` loop.

4.7 Traceroute Artifacts

Traceroute artifacts, such as bugs, outgoing interfaces, transient route changes, per-packet load balancing, etc., can result in errors or prevent inferences from being made. Even Paris traceroute, designed to avoid most types of load balancing, is not immune to artifacts [39].

Aside from interfaces that appear as third party addresses, (see § 4.4.3), there is a chance that artifacts in a $\mathbb{N}$ will cause MAP-IT to make an incorrect inference, either because the connected AS is wrong or because the interface is used internally. Using the neighbors sets is an attempt to diminish mistakes caused by artifacts but some errors are likely to occur, especially for IHs with small $\mathbb{N}$. As we show in § 5.3, higher values of f can further reduce the number of mistakes, at the expense of the recall. There is also the risk that artifacts in $\mathbb{N}$ could prevent MAP-IT from making an inference if they prevent an AS from appearing enough to induce an inference.

4.8 Stub AS Heuristic

The algorithm presented in the previous subsections assumes that for an inter-AS link, at least two interfaces from the connected AS will appear either before or after the link. However, the next hop after the link may always reply with the same interface address, due to using a NAT to connect to its ISPs, flow control, or when the number of probes (or the destinations used) is insufficient to expose additional interfaces. Many stub ASes fall into this category, so we provide a heuristic to infer links involving low visibility stub ASes. This heuristic is only invoked after trying to infer inter-AS links for all IHs with $\mathbb{N}$ containing more than one interface address.

Alg 4 presents the stub heuristic. We only consider forward links to stub ASes, under the assumption that if the link was named out of the stub AS's address space, a backward inference would have been made in a previous step, especially since the traces are moving from a more visible AS to a relatively less visible AS. To prevent invalid inferences, we ensure that there is no inference made for the other half of the interface, and that no backward inference was made for the neighboring IH. If all of the conditions are met, then MAP-IT infers an inter-AS link.

A potential source of error is that the neighboring interface may be the endpoint of the inter-AS link, instead of the inferred interface. We expect this to be unlikely because inferences are only made for stub ASes, suggesting a transit relationship from the first AS to the second AS. As noted before, links between providers and customers are typically assigned from the provider's address space. Additionally, if the neighbor is used for the inter-AS link, a backward inference should have been possible, because providers generally receive packets from many ingress points for their customers, likely exposing multiple interfaces on the border router. The absence of an inference suggests that the link is assigned addresses from the provider's address space.

Another risk is that there may be a hidden AS between the first AS and the stub AS, if there is a only a single hop in the hidden AS. This results in an invalid inference between disconnected ASes.

Importantly, for correctly identified stub and provider ASes, third party addresses will not cause a mistaken inference to be made in this step. A third party address returned by a stub AS will be one of its providers. By definition, the provider network is not a stub AS, so no inference will be made in that case.

4.9 Discussion

Unresponsive hops, unannounced addresses, third party addresses, and single interface neighbor lists for ISPs can cause MAP-IT to underestimate the number of inter-AS links. Additionally, interfaces used for public peering at IXPs can cause issues because they are often assigned addresses from a prefix shorter than `/30`, causing incorrect updates to the IP2AS mappings for the other side of the interface. We can fix some of the IXP related problems by incorporating known IXP prefixes from outside sources.

Another potential source of error is when ASes are owned by the same organization, called siblings [18]. MAP-IT relies on the same origin AS accounting for more than half of the interfaces before or after a border interface. Sometimes sibling ASes will appear in the same $\mathbb{N}$, but no single AS will appear frequently enough individually. We address this problem using CAIDA's AS2ORG tool [5], which is based on WHOIS information, to determine if two ASes are siblings. We also use this information to prevent the algorithm from inferring inter-AS links between siblings, as the borders between siblings are often blurry, and might have no practical implications. Unfortunately, WHOIS information alone is insufficient to identify all ASes that belong to the same organization, and the dataset might contain errors.

5. RESULTS

To validate MAP-IT, we ran our algorithm using traces collected by CAIDA's ARK infrastructure [6], which for the month of October, 2015 consisted of 110 monitors distributed throughout 43 countries [24]. For our experiments, we use the traces collected between October 1 and October 31, 2015, totaling 733,841,270 million traces. After discarding traces with interface cycles, we retain 714,027,556 traces with 6,565,421 distinct interface addresses, of which 4,992,879 appear adjacent to at least one other interface address.

For IP-to-AS mappings, we use BGP prefix announcements collected by 40 different monitors in October, 2015, which includes 18 monitors from RouteViews [11], 13 from RIPE RIS [3], and 9 from Internet2 [8], covering 30 distinct cities, 14 countries, and 6 continents. In using more than one route collector, we hope to see route announcements from a greater number of ASes [22]; prefixes that might be aggregated in specific geographic areas, possibly obscuring the originating AS; and to view prefixes that are not advertised in all geographic areas. We also use the Team Cymru IP2AS mapping tool [12] for prefixes not seen in the BGP announcements.

To help avoid problems caused by IXPs, we combine lists of IXP prefixes from PeeringDB [10] and Packet Clearing House (PCH) [9]. PeeringDB also provides IXP AS numbers for some of the IXPs, which we combine with the BGP announcements to identify additional IXP addresses. The IXP information is sometimes stale and incomplete, but is sufficient for our purposes.

The combination of BGP dumps, Team Cymru IP2AS mappings, special purpose/private prefixes, and IXP prefixes cover 99.2% of the usable interfaces seen in the traceroute dataset. To determine sibling ASes we use CAIDA's AS2ORG tool, as well as 140 additional pairs gathered from independent research. Mapping ASes to organizations is a difficult problem and some of the information may be incorrect, and other sibling pairs might be missing. Finally, we use CAIDA's AS Relationships dataset [4] to identify ISP ASes (ASes with at least one non-sibling customer). The AS relationship dataset is derived from BGP announcements and is prone to its own errors and incomplete relationship information.

5.1 Verification Datasets

We verify our inferences against two separate datasets. The first dataset is ground truth from Internet2's network (AS11537), which is a highly accurate, and continually updated, list of interfaces used on Internet2's IP backbone. The second verification is performed using DNS hostnames for interfaces in Level 3 (AS3356) and TeliaSonera (AS1299), from which we manually create the verification dataset. Using DNS hostnames allows us to test many more inferences than using only the Internet2 dataset. While they can be unreliable, in our experience they are accurate enough to be used as approximate ground truth.

5.1.1 Internet2 Verification

We first verify our results against the ground truth from Internet2 [7], a research and education (R&E) network that primarily connects other R&E networks to each other and to the global Internet. The data, which is available from Internet2 upon request, is an XML file that provides router and interface information for their IP backbone. Router information is broken down into interfaces, which contain information such as the subnet used to assign an address to the interface, and a label describing how they use the interface. From the interface descriptions included in the list, we manually determine which interfaces are used internally, and which are used for inter-AS links. The description also allows us to identify ASes connected by the inter-AS links; e.g., the description `UVM via AL2S/ALBA` indicates that its associated interface connects Internet2 to AS1351 (University of Vermont) using a VLAN across Internet2's Advanced Layer 2 Service.

The dataset is updated daily, protecting against stale information. In total, the Internet2 dataset covers 378 addresses seen in the traces, which includes 164 inter-AS links. Of those, 4 did not have any adjacent addresses from the connected ASes, so we did not include them.

5.1.2 Level 3 and TeliaSonera Verification

We also verify our inferences against the DNS hostnames assigned to interfaces in Level 3 and TeliaSonera. Both are global Tier 1 providers, and enable us to test MAP-IT on the commodity Internet. To create the dataset we tried to resolve the DNS hostnames associated with interface addresses in AS3356 and AS1299 seen in the traces, along with their inferred other side. If possible, we extract the hostnames from CAIDA's DNS hostname dataset [2], supplementing with additional hostname lookups performed on Nov 6, 2015. We use these networks because they often include a tag in their DNS hostnames associated with inter-AS link interfaces that indicate the connected network, usually by name.

After extracting the hostnames, we classified each hostname as either external, meaning that it contains an inter-AS link tag, or internal, which is indicated when the hostname lacks an inter-AS tag, and when the hostname associated with the other side of the interface lacks such a tag as well. Finally, we manually went through each external hostname and identified the AS connected by the inter-AS link. We identified 4645 inter-AS links – of which 4442 had adjacent addresses from the connected AS – and 3599 internal links.

An example of an external hostname is `cogent-ic-309423-den-b1.c.telia.net`, where `cogent-ic` indicates an interconnection between TeliaSonera and Cogent (AS174). On the other hand, the hostnames `ae-41-41.ebr1.berlin1.level3.net` and `ae-41-41.ebr2.budapest1.level3.net`, associated with the link interfaces `4.69.201.118` and `4.69.201.117` respectively, imply that the link connects two routers in Level 3's network, so both interfaces are classified as internal interfaces. We were not able to interpret every DNS hostname, such as `dialup` hostnames that do not include any information about the connected ISP network, or hostnames with ambiguous tags. When such hostnames are encountered, we remove the interfaces from our verification dataset. We also remove 176 interfaces with hostnames that appear to tag switching fabrics, such as the name of a data center, instead of the network connected by the VLAN.

There are two primary sources of noise. First, the hostname tags might be stale, and the interface is no longer used for the purpose indicated by its tag. Second, when corporations and networks change ownership, the hostname is not always updated to reflect that fact. We tried to track down the network's history in such cases but sometimes we could not. Both sources of noise inflate the number of false positives.

5.2 Precision and Recall

The evaluation against both datasets is mostly the same. A correct inference is one where MAP-IT correctly identifies an inter-AS link interface, and infers the ASes, or their sibling ASes, involved. Missing inferences are those where the algorithm failed to identify an inter-AS link indicated in the ground truth, with the qualification that the interface or its other side appears in the traceroute dataset. We also require that either the link is assigned a prefix from the connected AS, or that at least one address in the connected AS is seen adjacent to the link. We calculate precision as the fraction of correct inferences and recall as the fraction of inferred inter-AS links.

For both verification datasets, errors include inferences that failed to correctly identify the ASes that use the link and inferences made on internal interfaces. For Internet2, we also include any inference involving AS11537 on an interface not in the dataset[8]. Since we could not resolve DNS hostnames for all of the interfaces in AS3356 and AS1299, we cannot verify all inferences involving those two networks. Instead, for all inter-AS link interfaces in the datasets, we classify inferences involving the two ASes specified in the dataset as errors, if they were made on an adjacent interface in the connected AS. It is possible that some of these inferences are valid, but appear adjacent to the link due to artifacts, such as load balancing.

5.3 Selecting a Value For f

In § 4.4.1 we introduced a constant factor, f, which can be adjusted to increase either the precision or recall. Here, we evaluate this trade off for different choices of f. We ran experiments for different values of $0 \leq f \leq 1$ at increments of 0.1.

The results are shown in Figure 6. While the precisions for Level 3 and TeliaSonera remain mostly consistent for all values of f, it has a significant impact on the Internet2 results. For Internet2, there is improvement in the precision until $f = 0.5$, where it reaches 100%, and then drops sharply for $f = 0.9$ and $f = 1$. As MAP-IT requires more proof for each inference, the quality of the initial inferences is generally higher and it more quickly discards incorrect inferences as it refines the IP2AS mappings. However, for high values of f, MAP-IT is unable to refine the inferences because it is too constrained. For $f \geq 0.9$, when MAP-IT infers a link on a backbone link in Internet2, it is sometimes unable to discard it because there were insufficient updates to the IP2AS mappings.

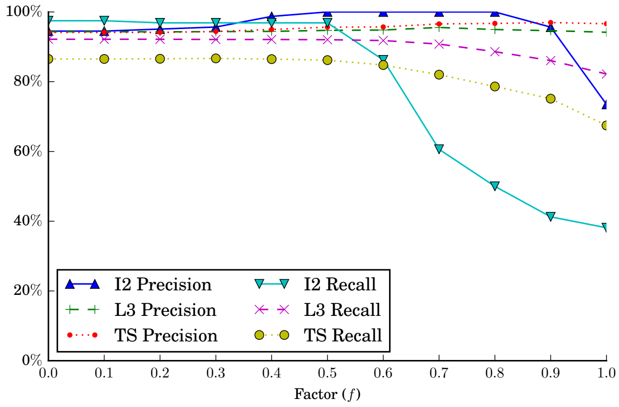


Figure 6: The impact of f.

Unlike precision, recall remains mostly constant for lower values of f, but sharply decreases for higher values. As explained above, the $\mathbb{N}$s for inter-AS link interfaces often have some addresses belonging to ASes not connected by the link. Increasing f can increase the certainty of each inference up to a point, but may prevent valid inferences due to the presence of those other addresses.

5.4 AS Relationships

Using results for $f = 0.5$ we also break down our results by the relationship type between the ASes inferred to share a link, shown in Tab 1. We use CAIDA's AS Relationship dataset [4] to classify the relationship between the networks as either a transit relationship (customer-provider) or a peering relationship, and to identify stub ASes. If an AS does not appear in the relationship dataset we classify the relationship as Stub Transit, and if there is no transit link between the ASes then we classify the relationship as Peer.

There is a dip in the precision between the Tier 1 networks and their peers. Of the 51 incorrect inferences, 31 are caused by inferences on an adjacent interface beyond the link. These inferences can both prevent MAP-IT from making a correct inference and are considered errors in our verification. We expect that some of these are valid links that appear adjacent to the peering in-

[8] We inferred two links involving AS11537, `198.71.46.44/31` and `64.57.28.30/31`, which are not in the dataset. We confirmed with the Internet2 NOC that they are correct inferences and that their exclusion from the dataset is an exceptional case. We do not mark those inferences as errors in our verification.

	TP	FP	FN	Precision%	Recall%
ISP Transit					
I2	25	0	2	100.0	92.6
L3	1170	63	170	94.9	87.3
TS	909	21	200	97.7	82.0
Peer					
I2	125	0	3	100.0	97.7
L3	121	22	17	84.6	87.7
TS	229	29	19	88.8	92.3
Stub Transit					
I2	5	0	0	100.0	100.0
L3	1221	55	30	95.7	97.6
TS	241	13	2	94.9	99.2
Total					
I2	155	0	5	100.0	96.9
L3	2512	140	217	94.7	92.0
TS	1379	63	221	95.6	86.2

Table 1: MAP-IT's inferences broken down by the relationship between the ASes.

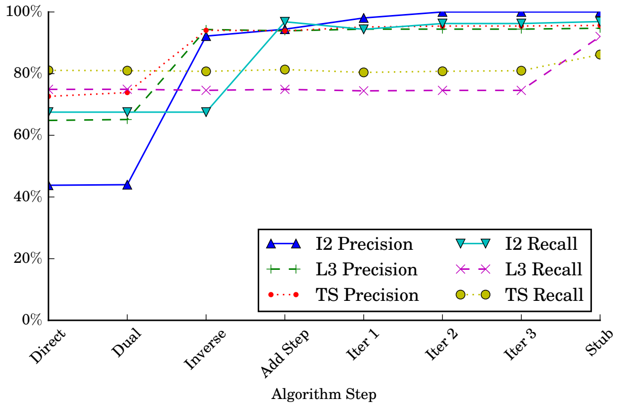


Figure 7: The impact of each step on the results.

terface due to traceroute artifacts. Not including these errors, the combined precision for Level 3 and TeliaSonera is 95.8% for peering links.

For both Level 3 and TeliaSonera there is a drop in the recall for links between these providers and other ISPs. This happens because sometimes following an inter-AS link only a single address is seen in the traces. When the address belongs to a stub AS, the stub heuristic will identify the link, but we do not trust a single address belonging to an ISP because it might be a third party address. One potential remedy is to try to expose more interface addresses by targeting the links with additional traces, which could enable more inferences.

As seen here, the absence of an inference does not imply that an interface is not used for an inter-AS link. This is especially true when the $\mathbb{N}_F$ or $\mathbb{N}_B$ is empty, contains a single interface, or contains primarily unannounced addresses, as all of these prevent the algorithm from making an inference due to insufficient evidence of an AS switch.

5.5 Utility of Multiple Passes

Next, we demonstrate the utility of the individual steps in the algorithm, as well as making multiple passes through the interfaces in order to refine the inferences. Fig 7 presents intermediate results after each part of the initial add step, and after each iteration, which includes an add step, followed by the remove step.

Initially, directly adding inferences based on the original IP-to-AS mappings performs reasonably well for Tier 1 networks but the precision for Internet2 is just 43.8%, showing the necessity of refining these inferences. After a slight improvement due to resolving point-to-point violations, removing inverse inferences brings the precision for all networks above 92%, indicating that inverse inferences are initially prevalent. Making additional passes through the interfaces in the first add step adds 46 correct inferences for Internet2, due to further refining the IP2AS mappings following each pass through the interfaces. The end of the first iteration brings additional improvements in the precision, removing 6 incorrect Internet2 inferences. The second iteration further refines the inferences, correcting three inferences for Internet2, adding two inferences for Level 3, and correcting four inferences for TeliaSonera. Finally, the stub heuristic vastly improves the recall for Level 3, which connects to many stub networks, with a less pronounced improvement for TeliaSonera.

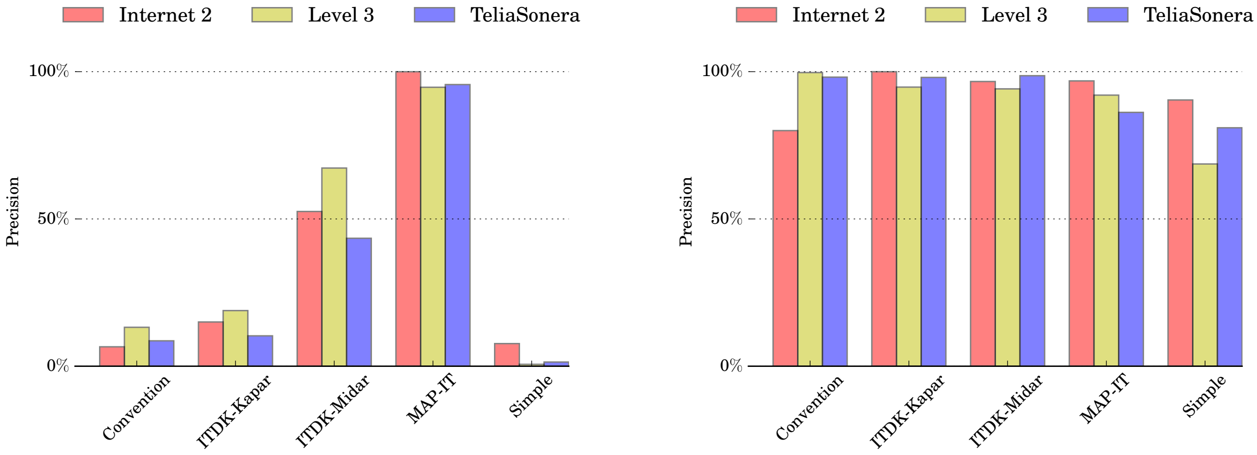
5.6 Comparison with Existing Approaches

Until bdrmap, there have been no verified attempts to identify the interfaces on inter-AS links, and there are no techniques that can be applied to existing traceroutes or that identify inter-AS links in ASes not directly connected to the traceroute monitor. Here, we compare MAP-IT with $f = 0.5$ to three techniques that are commonly considered sufficient for the purpose, which are the simple heuristic, the convention heuristic, and CAIDA's ITDK dataset. We discuss other potential comparisons in § 6 when discussing future work. The precision and recall are shown in Figures 8a and 8b respectively. Our verification demonstrates that MAP-IT outperforms these techniques for inferring inter-AS links, and shows they should not be relied upon to identify inter-AS link interfaces.

Simple Heuristic: In the first comparison, labeled Simple, we go through each trace looking for adjacent IP addresses in different ASes. The simple heuristic assumes that the first IP address in a different AS is used for the inter-AS link.

It can be tempting to think that this is a reasonable approach for identifying inter-AS link interfaces, but it is fraught with problems. Most notably, inter-AS links are assigned addresses from a common prefix, which this does not account for, leading to many incorrect inferences, and severely hurting the recall.

Convention Heuristic: The second heuristic, labeled Convention, is similar to the Simple approach, but incorporates the conventional wisdom that transit links are typically assigned addresses from the provider's ad-



(a)

(b)

Figure 8: Recall and precision of existing approaches vs. MAP-IT.

dress space. For each pair of adjacent IP addresses in different ASes, it first checks to see if one AS is a transit provider for the other based on the information in the AS relationship dataset [4]. If so, it assumes that the provider's address is used for the inter-AS link.

This helps better identify the interface address used for the inter-AS links in the Tier 1 networks. For Internet2, this causes incorrect and missed inferences because Internet2's transit links are often assigned a prefix from the customer's address space. Although this heuristic accounts for transit link addressing, there is no known heuristic for assigning addresses used on peering links, so it reverts to the Simple heuristic.

Both the Simple and Convention heuristics fail to account for traceroute artifacts, such as load balancing and third party addresses, which significantly reduces their precision. Additionally, these heuristics operate on the level of individual traces, which may cause them to infer many links for the same interface address.

CAIDA ITDK: The last approach is CAIDA's Internet Topology Data Kit (ITDK) [1], a publicly available dataset, which includes two router topologies. The first, labeled ITDK-Kapar uses `iffinder` [28], MIDAR [30], and `kapar` [29] to resolve the interfaces, called aliases, that reside on the same router. The second topology, labeled ITDK-MIDAR, does not use `kapar`.

In both topologies routers are assigned to ASes, and the links between routers are provided, along with the interface used on the link. ITDK-Kapar follows the methodology of Huffaker, et al. [21], while ITDK-MIDAR is similar to Giotsas, et al. [19], except that both do not use `iffinder`. The dataset is derived from traces on the ARK infrastructure between August 16 and August 29, 2015, so we reconstructed the verification datasets for the traces used to generate the ITDK dataset, using DNS information from this time period and Internet2 router information from August 25, 2015. We did not mark all links involving AS11537 that are not in the Internet2 dataset as errors, out of concern that the IP2AS mapping tool used for the ITDK datasets might map addresses in AS11164, a sibling AS, to AS11537.

Of the approaches, ITDK-MIDAR is the most accurate, yet the precision is only 52.2% for Internet2, 67.3% for Level 3, and 43.4% for TeliaSonera. If we remove the errors involving the adjacent addresses for Level 3 and TeliaSonera, the precision improves to 77.9% and 53.6% respectively. The low precision is caused by imperfect alias resolution and inaccurate router-to-AS mappings; router graphs are currently unable to accurately identify inter-AS link interfaces, despite their utility for other problem domains.

5.7 Impact of Artifacts

In addition to the recall and precision, we also provide anecdotal evidence that the algorithm is resilient to a small amount of traceroute artifacts.

An example we found during our experiment is the interface `4.68.110.186`, which is announced by AS3356. The hostname, `mci-level3-ae.chicago3.level3.net`, indicates that it is an interface used to connect with AS701 (Verizon, parent of MCI). Our algorithm classifies it as a inter-AS link interface between AS3356 and AS701 because 113 of the addresses in its $\mathbb{N}_F$ belong to AS701 out of a total of 141 interface addresses.

Interestingly, 5 of the remaining interfaces belong to prefixes announced by AS3356, which is most likely due to transient routing changes or load balancing along a path that uses this link. Due to the overwhelming evidence that this is an inter-AS link interface, our heuristic is able to look past the anomalous traceroute results and make the correct inference.

6. CONCLUSION AND FUTURE WORK

MAP-IT is a novel multi-pass algorithm for precisely inferring inter-AS link interfaces. Using two separate datasets, MAP-IT's precision ranges between 94.7% and 100.0%, with recalls between 86.2% and 96.9%. These results show that MAP-IT may be a useful tool for network operators and researchers concerned with accurately mapping links crossing AS boundaries.

The implementation used for our experiments is available at `http://www.seas.upenn.edu/~amarder/aslinks.html`. It can be run using existing or custom traceroute datasets.

At the time of writing we were unaware of the work by Giotsas, et al. [19]. They use inter-AS links derived from a router-level graph built with MIDAR as inputs to their Constrained Facility Search, which iteratively refines the possible interconnection facilities for inferred peerings. Their technique for identifying inter-AS link interfaces is similar to ITDK-MIDAR, and comparing our results with theirs is interesting future work. The same is true for `bdrmap`, whose performance suggests that head-to-head comparisons with MAP-IT in a variety of contexts would be fruitful. It would also be interesting to evaluate the impact of incorporating AS-level path alterations, such as those performed by Chen, et al. [16].

Acknowledgments

This work was partially supported by DARPA under Contract numbers PO-0004103, HR0011-16-C-0061, and FA8650-11-C-7189. Computational analysis was performed using a high-performance Unified Computing System cluster provided by Cisco Systems; we thank them for their generosity. Operators at GlobalNOC and Internet2 assisted us in generating the Internet2 dataset. We thank the referees for their deep and constructive comments, and appreciate the insight of the IMC PC Chairs in encouraging sharing submitted versions of papers with [33]. We especially thank our shepherd, Walter Willinger for his help and insight.

7. REFERENCES

[1] Internet Topology Data Kit - August 2015. `http://www.caida.org/data/internet-topology-data-kit/`.

[2] IPv4 Routed /24 DNS Names Dataset– Sep 24, 2015 to Nov 7, 2015. `http://www.caida.org/data/active/ipv4_dnsnames_dataset.xml`.

[3] RIPE RIS Raw Data. `https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris/ris-raw-data`, Oct 2015.

[4] The CAIDA AS Relationships Dataset, October 2015. `http://www.caida.org/data/as-relationships/`.

[5] The CAIDA UCSD AS to Organization Mapping Dataset, July 7, 2015. `http://www.caida.org/data/as_organizations.xml`.

[6] The CAIDA UCSD IPv4 Routed /24 Topology Dataset – Oct 1, 2015 to Oct 31, 2015. `http://www.caida.org/data/active/ipv4_routed_24_topology_dataset.xml`.

[7] Internet2 - Interface Addresses. `http://vn.grnoc.iu.edu/Internet2/interfaces/interfaces-addresses.html`, Oct 2015.

[8] Internet2 NOC BGP Rib Dumps. `http://ndb7.net.internet2.edu/bgp/`, Oct 2015.

[9] Packet Clearing House: Internet Exchange Directory. `https://prefix.pch.net/applications/ixpdir/menu_download.php`, Oct 2015.

[10] PeeringDB. `https://beta.peeringdb.com/`, Oct 2015.

[11] University of oregon route views project. `http://www.routeviews.org/`, Oct 2015.

[12] IP to ASN Mapping. `http://www.team-cymru.org/IP-ASN-mapping.html`, Aug 2016.

[13] A. Bender, R. Sherwood, and N. Spring. Fixing ally's Growing Pains with Velocity Modeling. In *Proc. IMC*, pages 337–342. ACM, 2008.

[14] J. Chabarek and P. Barford. What's in a name?: decoding router interface names. In *Proc. of the 5th ACM HotPlanet*, pages 3–8. ACM, 2013.

[15] H. Chang, S. Jamin, and W. Willinger. Inferring AS-level Internet Topology from Router-Level Path Traces. In *Proc. ITCom 2001*, pages 196–207. International Society for Optics and Photonics, 2001.

[16] K. Chen, D. R. Choffnes, R. Potharaju, Y. Chen, F. E. Bustamante, D. Pei, and Y. Zhao. Where the Sidewalk Ends: Extending the Internet As Graph Using Traceroutes from P2P Users. In *Proc. ACN CoNEXT*, CoNEXT '09, pages 217–228, New York, NY, USA, 2009. ACM.

[17] M. Cotton, L. Vegoda, R. P. Bonica, and B. Haberman. Special-Purpose Address Registries. RFC 6890, RFC Editor, April 2013.

[18] L. Gao. On Inferring Autonomous System Relationships in the Internet. *IEEE/ACM ToN*, 9(6):733–745, 2001.

[19] V. Giotsas, G. Smaragdakis, B. Huffaker, M. Luckie, and k. claffy. Mapping Peering Interconnections to a Facility. In *Proc. ACM CoNEXT*, Dec 2015.

[20] R. Govindan and H. Tangmunarunkit. Heuristics for Internet Map Discovery. In *Proc. INFOCOM 2000*, volume 3, pages 1371–1380. IEEE, 2000.

[21] B. Huffaker, A. Dhamdhere, M. Fomenkov, and k. claffy. Toward Topology Dualism: Improving the Accuracy of AS Annotations for Routers. In *PAM*, Zurich, Switzerland, Apr 2010. PAM 2010.

[22] B. Huffaker, M. Fomenkov, and k. claffy. Internet Topology Data Comparison. Technical report, Cooperative Association for Internet Data Analysis (CAIDA), May 2012.

[23] B. Huffaker, M. Fomenkov, and k. claffy.

DRoP:DNS-based Router Positioning. *ACM SIGCOMM CCR*, 44(3):6–13, Jul 2014.

[24] Y. Hyun. Ark update: Present and future. AIMS 2015 Workshop, Mar 2015.

[25] Y. Hyun, A. Broido, and k. claffy. On Third-party Addresses in Traceroute Paths. In *PAM*, San Diego, CA, Apr 2003. PAM.

[26] V. Jacobson. *traceroute(8) FreeBSD System Manager's Manual*, May 2015.

[27] M. S. Kang and V. D. Gligor. Routing Bottlenecks in the Internet: Causes, Exploits, and Countermeasures. In *Proc. CCS*, pages 321–333. ACM, 2014.

[28] K. Keys. iffinder. `https://www.caida.org/tools/measurement/iffinder/`.

[29] K. Keys. Internet-scale IP alias resolution techniques. *ACM SIGCOMM CCR*, 40(1):50–55, 2010.

[30] K. Keys, Y. Hyun, M. Luckie, and k. claffy. Internet-Scale IPv4 Alias Resolution with MIDAR. *IEEE/ACM ToN*, 21(2):383–399, Apr 2013.

[31] M. Luckie and k. claffy. A Second Look at Detecting Third-Party Addresses in Traceroute Traces with the IP Timestamp Option. In M. Faloutsos and A. Kuzmanovic, editors, *PAM*, volume 8362 of *Lecture Notes in Computer Science*, pages 46–55. Springer International Publishing, 2014.

[32] M. Luckie, A. Dhamdhere, D. Clark, B. Huffaker, and k. claffy. Challenges in Inferring Internet Interdomain Congestion. In *Proc. IMC*, IMC '14, pages 15–22, New York, NY, USA, 2014. ACM.

[33] M. Luckie, A. Dhamdhere, B. Huffaker, D. Clark, and k. claffy. bdrmap: Inference of Borders Between IP Networks. In *Proc IMC*, IMC '16, New York, NY, USA, 2016. ACM.

[34] Z. M. Mao, D. Johnson, J. Rexford, J. Wang, and R. Katz. Scalable and Accurate Identification of AS-Level Forwarding Paths. In *Proc. IEEE INFOCOM 2004*, volume 3, pages 1605–1615. IEEE, 2004.

[35] Z. M. Mao, J. Rexford, J. Wang, and R. H. Katz. Towards an Accurate AS-Level Traceroute Tool. In *Proc. SIGCOMM*, pages 365–378. ACM, 2003.

[36] P. Marchetta, W. de Donato, and A. Pescapé. Detecting Third-Party Addresses in Traceroute Traces with IP Timestamp Option. In *PAM*, pages 21–30. Springer, 2013.

[37] A. Retana, R. White, V. Fuller, and D. McPherson. Using 31-Bit Prefixes on IPv4 Point-to-Point Links. RFC 3021, RFC Editor, December 2000.

[38] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP Topologies with Rocketfuel. In *ACM SIGCOMM CCR*, volume 32, pages 133–145. ACM, 2002.

[39] F. Viger, B. Augustin, X. Cuvellier, C. Magnien, M. Latapy, T. Friedman, and R. Teixeira. Detection, Understanding, and Prevention of Traceroute Measurement Artifacts. *Computer Networks*, 52(5):998–1018, 2008.

[40] B. Zhang, J. Bi, Y. Wang, Y. Zhang, and J. Wu. Refining IP-to-AS mappings for AS-level traceroute. In *Proc. ICCCN*, pages 1–7. IEEE, 2013.

[41] M. Zhang, Y. Ruan, V. S. Pai, and J. Rexford. How DNS Misnaming Distorts Internet Topology Mapping. In *USENIX Annual Technical Conference, General Track*, pages 369–374, 2006.

[42] Y. Zhang, R. Oliveira, Y. Wang, S. Su, B. Zhang, J. Bi, H. Zhang, and L. Zhang. A Framework to Quantify the Pitfalls of Using Traceroute in AS-Level Topology Measurement. *Selected Areas in Communications, IEEE Journal on*, 29(9):1822–1836, 2011.

[43] Y. Zhang, R. Oliveira, H. Zhang, and L. Zhang. Quantifying the pitfalls of traceroute in AS connectivity inference. In *PAM*, pages 91–100. Springer, 2010.