

Efficient AS Network Topology Measurement Based on Ingress to Subnet Reachability

Ibrahim Ethem Coskun
University of Nevada, Reno
ethemcoskun@nevada.unr.edu

Muhammed Abdullah Canbaz
University of Nevada, Reno
mcanbaz@unr.edu

Mehmet Hadi Gunes
University of Nevada, Reno
mgunes@unr.edu

Abstract—Internet topology measurement studies rely on a few vantage points to trace towards a large number of destinations. Such traces contain significant overlaps, which increase the measurement traffic and duration. As there is considerable redundancy in the generated measurement probes, researchers have been exploring probe reduction approaches with minimal loss of the topological information. To decrease the probing overhead, in this paper, we propose ingress based probing technique which relies on the ingress points of an Autonomous System (AS) towards the target subnetworks. To identify ingress points for each subnetwork of the target AS, we first probe the subnet prefixes announced via BGP. After identifying the ingress points of the subnetworks, the subsequent probing is conducted based on the vantage point reachability of the ingresses. Additionally, we present how to utilize subnets with the tree based probe reduction approaches. As shown in an experimental study with 10 ASes, ingress based approaches considerably reduce the probing overhead while minimizing the loss of topological information.

I. INTRODUCTION

Internet consists of tens of thousands of distinct ASes interconnected with each other. As each AS has its own network deployment practices and routing policies, the characteristics of the overall network is an open question that researchers have been tackling with [2]. Network measurement helps us better understand the characteristics of the Internet and improve Internet topology and services along with management of large scale ISPs [3]. Internet measurement is concerned with both topology [2] and traffic [1]. Internet topology measurement involves active measurements where probe packets are used to collect information about network devices. In general, a large number of traces are used to collect path information about paths from vantage points to a given set of destinations. It is important to have a good vantage point distribution over the Internet to obtain a non-biased sample topology [4].

One of the challenges faced when mapping a network is the significantly redundant trace traffic [5]. Minimizing the redundant probing is also important in terms of the time it takes to complete a measurement campaign as well as the time for processing the collected data. High volumes of probing traffic towards a destination AS might trigger firewalls and be blocked by the destination AS [6]. Thereby, minimizing redundant probes is important not only in the sense of efficiency but also to abstain from involuntary disruption of the measured networks.

In this paper, we present an AS topology measurement approach, named *InSub*, that focuses on identifying the ingress

points of the announced subnet prefixes of the AS with respect to available vantage points. In the *InSub* approach, we first identify the ingress points of the destination AS for each subnet prefix announced via BGP from all vantage points. Then, we probe known subnet IP addresses through each ingress by selecting a subset of the vantage points. The proposed *InSub* approach considerably diminishes the redundant probing as the vantage points that probe a subnet prefix through the same ingress are eliminated. In our evaluations, we observe that *InSub* approach reduces the number of probes in the destination AS by more than half. Additionally, we present subnet based tree reduction approach, named *SubnetTree*, where subnets of IP addresses are taken into account while performing the tree reduction approach of [5]. As *InSub* approach is orthogonal to the tree based probe reduction, we can combine both approaches, named *InSubTree*, to further reduce the number of probes.

In the rest of the paper, we first present related work in Section II. We present the *InSub* probe reduction approach that utilizes ingress to subnet prefix reachability of ASes in Section III. We then present experimental results in Section IV and conclude with Section V.

II. RELATED WORK

In this section, we provide a brief overview of current approaches to network probe overhead reduction.

Rocketfuel was one of the earlier systems implementing a probe reduction approach, named *directed probing* [7]. Directed probing is based on recognizing transit traceroutes that pass through a network by identifying the entry and exit points of the traces based on BGP announcements collected from the RouteViews [8]. As **Rocketfuel** tries to minimize redundant crossing of an AS based on the BGP information, it misses the actual link level connectivity and hence potential topological views.

Scriptroute handles inter-monitor redundancy by *reverse path tree* approach, which considers the tree built around a destination node when probed from multiple vantage points [9]. When a trace towards a destination is completed by one of the vantage points, the information about the visited IPs in the trace is embedded in the measurement script of other vantage points. If the script recognizes that it has reached part of an already traversed path, then it stops. However, this system does not deal with intra-monitor redundancy.

Doubletree approach observes the tree-like structure of Internet routes around vantage points and destinations and deals with both intra- and inter- monitor redundancies [5]. Doubletree first decides a tuning parameter to estimate mid point for a path. Then, a vantage point probes both forwards toward the destination and backwards toward the vantage point starting from the mid point. Each vantage point adds the traversed paths to a local and a global stop sets. However, as the vantage points need to share information about the visited paths, this approach involves control traffic between the vantage points which adds to the communication overhead and adds delay to the measurement campaign.

Ingress point spreading approach integrates Subnet Centric Probing, Interface Set Cover, and Vantage Point Spreading [10]. Subnet Centric Probing utilizes subnetting of IPs and expands the Doubletree by eliminating the mid point parameter. By using notional prefix ingresses and hop number of the first router interface that followed by an IP within the target prefix, IPS computes a per-destination rank ordered list of vantage points. After selection of the vantage points, IPS probes the destination IPs by spreading the targets among the vantage points. Similar to Doubletree, however, the approach involves coordination traffic among vantage points.

Cheleby deals with intra- and inter- monitor redundancies by collecting partial traces from the ingress points of destination ASes and tracing each destination from each of the geographically clustered vantage points [2]. The approach reduces redundant probes without coordination traffic between vantage points.

The proposed InSub approach, extends the *ingress point spreading* approach so that it does not involve coordination traffic among vantage points similar to the *Cheleby* system. We also show how subnet information can be utilized with the tree approach so that probes are further reduced. Note that, ingress based probing can be utilized to efficiently map a destination AS. In that sense, our approach differs from the earlier methods that do not focus on a particular AS.

III. INGRESS TO SUBNET REACHABILITY BASED PROBE REDUCTION

In this section, we present InSub probe reduction approach which is based on the detection of ingress points of subnet prefixes with respect to the vantage points. InSub probe reduction approach focuses on two essential information. First, it discovers the ingress points of subnet prefixes announced by the destination AS. Second, it efficiently distributes the probing of subnet IPs to the vantage points based on the identified ingress points of the subnetworks.

A. Ingress Point Identification (IPI)

We first identify ingress points of subnet prefixes from each vantage point. To this end, we gather BGP announcements (e.g., from RIPE-NCC [11]) and filter the data by removing the overlapping smaller sized subnet prefixes (i.e., with larger CIDR). If there are multiple overlapping prefixes announced

by an AS, we only use the larger subnet announcement to identify the ingresses.

When tracing toward a destination AS, probes to the preceding ASes are typically redundant as they overlap in their path to reach the destination AS. Hence, in order to address intra-monitor redundancy, we minimize probes to routers outside the destination AS similar to the Cheleby approach that sets the initial TTL of traces based on the previous ones toward the AS [2]. Starting path traces from a certain hop distance from the vantage point tries to direct probes towards network regions that would be within the destination AS.

For each vantage point, we initialize the first hop of the path trace towards a target IP starting from TTL_{min} , which is typically 1 but can be set to a higher value to bypass internal routers. We then set the initial TTL value of the rest of the path traces towards the other IPs of the subnet prefix based on the hop distance H_i of the ingress point that the first trace revealed. If any of the subsequent traces to the AS have an ingress hop distance H_i greater than the previously set TTL, then we do not change the initial TTL value. However, if any of the ingress hop distance H_i of subsequent traces to the AS is smaller than the previously set TTL value, then we probe backwards until a new ingress point is identified, and update the initial TTL value of subsequent traces to H_i .

For example, assume we have a network as shown in Figure 1. We send the first trace to Target IP1, and identify the ingress point at the 5th hop from the vantage point VP. We send the second trace to IP2 with the initial TTL value of 5 (i.e., $H_i = 5$). The second trace starts from router 'A' while the ingress point for IP2 is at the 6th hop. In this case, TTL value is kept at 5. When we start the third trace from the 5th hop at route B, we cannot observe the ingress point. Hence, we probe backwards toward the VP until the ingress point is observed at hop distance 3. Hence, we update the initial TTL value for the rest of the traces to 3 (i.e., $H_i = 3$). Finally, the fourth trace to IP4 starts from 3rd hop at router 'C'.

In order to determine the ingress point of a subnet prefix from a vantage point, we probe the first and the last IPs in each announced subnet prefix of the destination AS. We then compare the identified ingress points of the first and the last IPs. If both traces has the same ingress point, then it is highly likely that all of the IPs in the subnet prefix will reach the

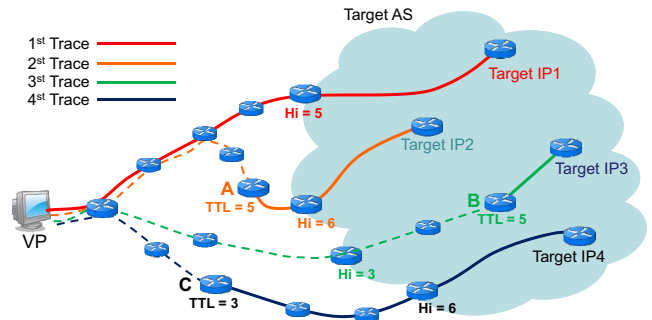


Fig. 1: Ingress point and hop distance determination

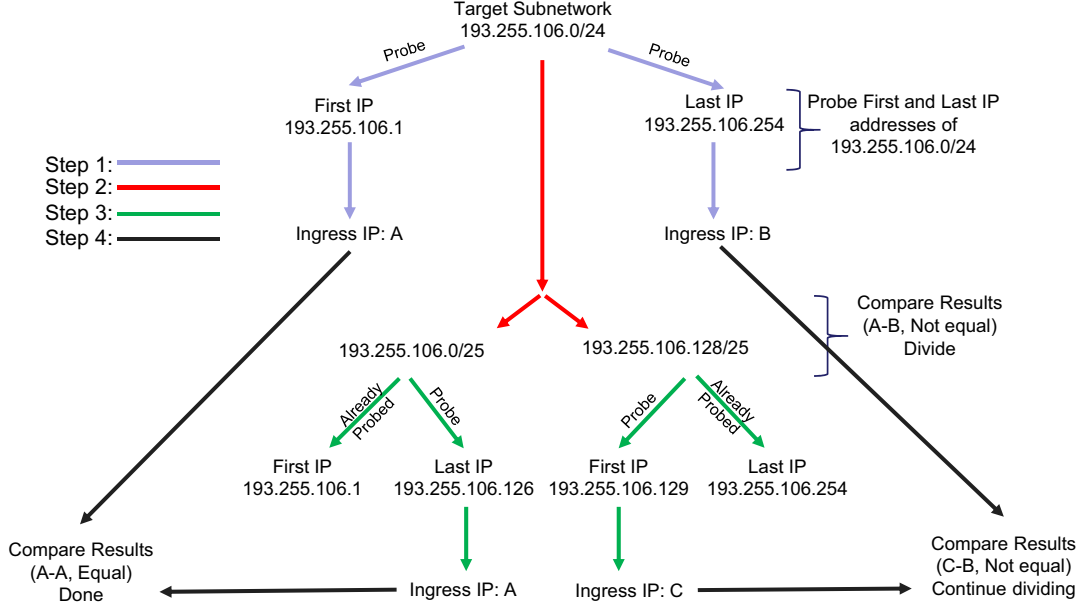


Fig. 2: Ingress point identification of subnet prefixes

target AS through the same ingress. However, if the ingress point of the two traces differ, we then divide the subnet prefix in the middle and probe each new subnet prefix separately. As the first IP of the first subnet prefix is already probed, we probe the last IP of the first subnet prefix to determine its ingress point. Similarly, we only probe the first IP of the second subnet prefix. We recursively continue to divide and probe the subnet prefixes until ingress points of both ends match or the ingress points are unreachable. Note that, due to various reasons, traces toward an IP might not reach any IP of the destination AS. In such instances, we can not determine the ingress point of the subnet and discard such subnet prefixes.

For example, consider a subnet address of 193.255.106.0/24 as shown in Figure 2. As the first IP, i.e., 193.255.106.1, and the last IP, i.e., 193.255.106.254, of the subnet enter the target AS from different ingress points, i.e., A and B, respectively, we split the subnet into two as 193.255.106.0/25 and 193.255.106.128/25. Then, we recursively determine the ingress points of both subnets. In the example, the last IP of the first subnet, i.e., 193.255.106.126, also enters the destination AS from the ingress point A. Hence, 193.255.106.0/25 subnet is marked to have an ingress point of A. However, as the first IP of the second subnet, i.e., 193.255.106.129, enters the destination AS from ingress point C, 193.255.106.128/25 subnet is further divided into two subnets as 193.255.106.128/26 and 193.255.106.192/26 to be further probed.

B. Subnet based Tracing through an Ingress (InSub)

In order to minimize inter-monitor redundancy, we assign vantage points to trace subnet IPs based on the identified ingress points of the subnet. To this end, we determine the optimal vantage point set to trace subnet prefixes of the

destination AS. Different from [12], vantage points do not communicate with each other but simply obtain probing tasks from the server and report the results back. Coordination based approaches require additional coordination traffic to exchange probing results among vantage points that often have limited capability.

After identification of ingress points of subnets with respect to the vantage points, we determine vantage points that will trace through each ingress point. Different edges of a network can be identified if it is traced from different ingress points. Ideally, we would like to trace every subnet prefix via each of the ingresses. However, that is not possible due to the peering relation between ASes. In particular, when an AS announces a subnet prefix only through certain egress points, the subnet is unreachable through other ingress points of the AS.

Additionally, we want to evenly distribute the probing workload of path tracing to all of the vantage points that can trace through each ingress point. Note that there are two optimization problems, (i) to evenly distribute *workload among all vantage points that can reach a particular ingress* and (ii) to evenly distribute *workload among all vantage points*. The optimization of vantage point load distribution becomes additionally challenging due to the unstable vantage points and varying probe times of destination addresses. Note that the server monitors vantage point status as vantage points such as PlanetLab are known to be unsustainable.

InSub generates a table of each ingress-subnet pair and their corresponding vantage points based on the previous subnet ingress identification mechanism. When we assign a subnet prefix probing task to a vantage point, we consider the subnet's coverage through the ingress and vantage points' distance to the ingress. We start dispatching probe assignments to vantage

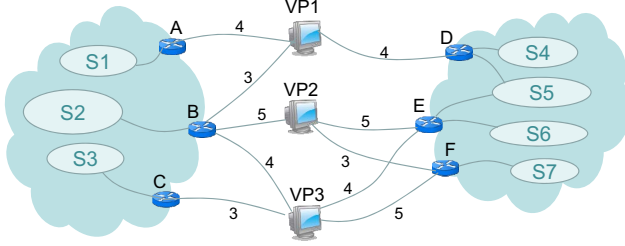


Fig. 3: Vantage point assignment for a sample topology

TABLE I: Potential vantage points for Ingress-Subnet pairs

A-S1	B-S2	C-S3	D-S4	D-S5	E-S5	E-S6	F-S7	VP1	VP2	VP3
VP1, 4	VP1, 3	VP3, 3	VP1, 4	VP1, 4	VP3, 4	VP3, 4	VP2, 3	A-S1	F-S7	C-S3
	VP3, 4				VP2, 5	VP2, 5	VP3, 5	D-S4		
	VP2, 5							D-S5		

points by examining the ingress-subnet pairs that are covered by only one vantage point. After all such ingress-subnet pairs are assigned to their respective vantage point, remaining idle vantage points are assigned ingress-subnet pairs that are least covered. We dynamically assign probes so that vantage points do not become idle while others have considerable probing load to perform. We prioritize the tracing of ingress-subnet pairs that can be probed by fewer vantage points to ensure that they are probed before the subnets that are reachable by more vantage points. If there are multiple ingress-subnet pairs with the same number of vantage point coverage, we select the one that is closest to the vantage point in question. This choice reduces probing time as vantage points closer to the ingress are expected to have a smaller RTT. If there are still multiple ingress-subnet pairs at the same hop distance to the vantage point, we select the subnet that has the largest potential workload where workload is estimated by the number of destination IPs in the subnet.

Figure 3 presents a sample of vantage point reachability of ingress-subnet pairs previously gathered from the IPI phase. In this example, there are seven subnets (i.e., S1, S2, S3, S4, S5, S6, and S7) and three vantage points (i.e., VP1, VP2, and VP3). First, the system generates an ingress-subnet accessibility table as shown in Table I. Subnets S1, S2, S3, S4, S6, and S7 can be reached by ingress points A, B, C, D, {D, E}, E, and F, respectively. Vantage point VP1 reaches subnet S1 through ingress point A in 4 hops, subnet S2 through ingress point B in 3 hops, and subnets S4 and S5 through ingress point D in 4 hops.

After the ingress-subnet accessibility table is built, we assign vantage points to ingress-subnet pairs starting with the least covered ingress-subnet pair. In the example, the least covered ingress-subnet pairs are A-S1, C-S3, D-S4, and D-S5 having only 1 VP in their coverage set. Hence, these pairs are first assigned to their only vantage point, resulting with VP1 with three tasks (i.e., A-S1, D-S4, D-S5) and VP3 with one task (i.e., C-S3) as shown in Table I. As only VP2 is left without a task, we pick least covered ingress-subnet pair among the ones that can be performed by VP2. In this case,

E-S5, E-S6, and F-S7 ingress-subnet pairs have two potential vantage points and V2 is closest to F-S7 among them. Hence, V2 is assigned F-S7. The system will then wait for one of the vantage points to become free to assign one of the remaining ingress-subnet pair.

After assigning vantage points to the ingress-subnet pairs of a destination AS, we trace IPs of the subnet prefix starting from the hop distance h of the ingress point. As network reachability changes over the time, we need to determine the ingress information right before network mapping. Moreover, the ingress point of an IP might differ from previously identified ingress point of the subnet due to topological or traffic changes. In such cases, we update the ingress point of the subnet to the newly identified one. If we observe the ingress point of an IP to be at a higher hop distance, we would have redundant probes. However, if our initial hop is inside the destination network, we need to backtrace until the ingress point is identified.

C. Tree based Tracing through an Ingress (SubnetTree)

As routing algorithms are based on (weighted) shortest paths, path traces from an ingress lead to tree like graphs. To reduce redundant probing due to traces that overlap, we can adopt the Doubletree [5] approach. Different from Doubletree, however, we do not coordinate between vantage points to stop traces. Instead, each vantage point observes the tree-like graphs from each ingress point that it is assigned.

For each subnet prefix, we probe the first IP starting from the previously identified ingress point. If the destination IP is reached at a hop distance of h , then we perform backward tracing for the rest of the IPs in the subnet starting from the hop distance of h . Assume, we have a trace from ingress point at distance h_I to an IP IP_D of a subnet S , and that the trace concluded at a hop distance of h_D . For subsequent IPs IP_S of the subnet S , we start the backward traces with a hop distance of $h_S = h_D$ until we observe an IP from any of the previous traces. Whenever we see an IP IP_x that was previously observed, we stop the trace. Additionally, if we do not observe the target IP IP_S at distance h_S , then we probe forward to reach the target IP IP_S . If the trace reaches the target IP IP_S or return 5 consecutive unresponsive routers, we stop the trace. Note that the likelihood of a responsive router after 5 unresponsive routers is very low [13]. If the IP_S is observed at a greater hop distance, then h_S is updated to the new distance.

Figure 4 presents a sample graph to demonstrate the tree based tracing of destination subnets through an ingress point. In the sample graph, there are two subnets that will be traced. Assume we start with a trace toward D. In this case, we will start from hop distance of 5 from the ingress and increment TTL values until we reach the destination D and obtain a trace of A-B-C-D. Trace to the next destination E of the subnet will start from the hop distance of 8 (i.e., distance of D) and backtrack until it merges with the tree. In this case, we will simply have E-C trace as backward tracing will stop at C, an IP that has already been observed. Trace to the IP C of the subnet will simply be ignored as C has already been observed.

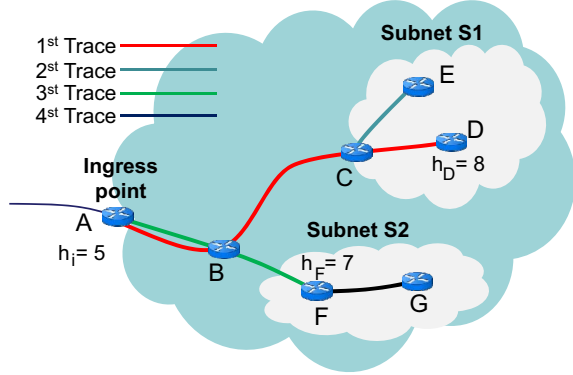


Fig. 4: Sample topology for tree based tracing

Additionally, trace to the IP F of second subnet will start from ingress point at hop distance 5 and reveal a path of A-B-F. As F was at hop distance of 7, the next destination of the subnet (i.e., G), will be traced starting from hop distance of 7. As 7th hop would reveal F, which is part of the tree, backward tracing will stop. However, as G has not been reached, forward tracing will reveal F-G path. As G was at 8th hop, any other IPs of the subnet would have been traced backward starting from a TTL of 8.

IV. EXPERIMENTAL RESULTS

We perform our measurements with Planetlab monitors around the world [14]. Because Planetlab nodes are not stable and some of the nodes did not support the required code, on average, we could only utilize around 200 of PlanetLab nodes as our vantage points to perform the measurements. We relied on RIPE-NCC [11] for AS prefix announcements and on ANT Census’s Internet Address Census Data [15] for determining destination IP addresses of ASes.

We collected traces using Paris traceroute to minimize the effect of load balancing on individual path traces [16]. We set the probing algorithm to ‘hopbyhop’ that enables all packets sent to a destination point to hold the same flow-identifier so that it reveals accurate paths when there is a load balancing router in the target network. We also set the return flow-id of the probes using (-r) flag. However, it is not always possible to accurately control the return flow identifier of the probes [17]. We set the ‘-w’ flag to 500 milliseconds as it was shown that response time of over 99.9% of the responsive nodes is 500 milliseconds [2]. We also chose ICMP as our probing protocol as it provides the highest responsiveness compared to TCP and UDP based probes [18].

We selected 10 medium sized ASes with a high customer cone based on the CAIDA ranking of ASes by customer cone size [19]. Table II shows the characteristics of the ASes selected for mapping. AS peers indicate the number of neighboring ASes as identified from BGP announcements and traceroutes; IPv4 prefixes and IPv4 addresses are based on the BGP announcements of ASes as collected by CAIDA. Additionally, probed IPs are obtained from the ANT Census data

TABLE II: Measured Autonomous Systems.

AS Number	AS rank	AS peers	Subnet Prefixes	IPv4 addresses	Probed IPs
174	2	4 768	134 631	648 411 904	110 411
1273	11	367	34 978	147 194 624	97 017
6762	7	370	82 712	238 719 232	85 065
2828	9	1 121	49 541	249 949 784	61 900
1239	20	650	35 281	329 687 040	61 059
3356	1	4 239	190 138	715 498 496	57 827
6939	8	3 980	74 594	305 295 872	39 847
6461	13	1 439	26 306	106 985 984	30 553
3491	12	569	50 952	184 550 656	26 054
6453	6	674	108 546	447 639 040	22 553

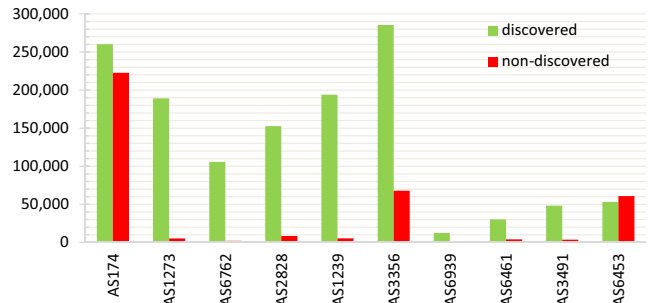


Fig. 5: Probed subnet prefixes per AS

and the IPs observed during the ingress point identification. Note that presented tables and figures are sorted based on the probed IPs of each AS.

A. Ingress Point Identification (IPI) Phase

In this section, we present measurement results of the Ingress Point Identification (IPI) phase with the selected ASes.

Figure 5 presents the subnet prefixes that were identified based on the IPI phase. Green bars indicate the number of subnets for which an ingress point was successfully identified while red bars indicate subnets whose ingress points were not discovered. The identified subnet prefixes are more than the announced IP prefixes as we split a prefix if it is observed via different ingresses from any of the vantage point. We obtained 2.5 times more subnet prefixes on average of all ASes with only AS6939 having fewer subnets than announced prefixes. We observe a large number of prefixes for which we could not identify an ingress. As prefixes are split recursively to determine the ingress points, sub-prefixes that do not reveal an ingress point for an edge IP is marked as non-discovered. Note that there were overlaps in announced prefixes and we ignored smaller ones when determining initial subnet prefixes. Additionally, the identified subnet prefixes might not correspond to a single physical link but rather it indicates that we observe the subnet-prefix to enter through the same ingress from the utilized vantage point.

Figure 6a presents destinations selected from end-points of subnets during the IPI. Green bars indicate the traces that reached the targeted IP while red bars indicate the ones that did not. As destination IPs are chosen as the end points of subnets, they might not correspond to a real system. Hence,

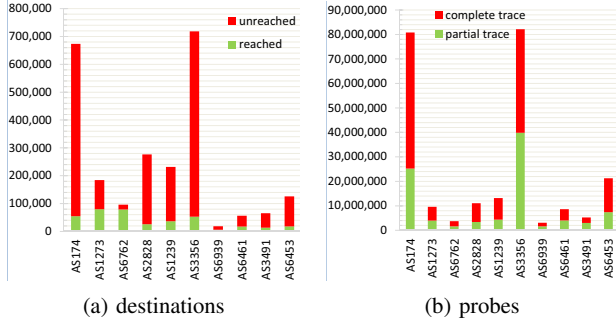


Fig. 6: Probing for ingress point identification

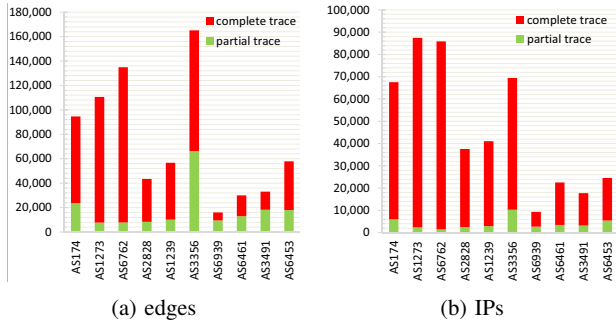


Fig. 7: Ingress point identification phase discovery

there is a much larger number of IPs that could not be reached. On average for all ASes, only 26.5% of traced IPs of ASes were reached while, in total, only 15.7% of 2 447 249 traced IPs were reached. Five ASes with largest number of non-discovered subnets (red bars in Figure 5) had the lowest reachability rates as well. Note that, however, even though an IP might not be reached, the trace towards that IP can still reveal the ingress point of the subnet prefix.

Figure 6b presents the number of probes that were sent to each AS during the IPI phase. Note that different from commonly used 3 probes per hop, we sent 2 probes per hop during the ingress identification. Also, we eliminate redundant probing by starting traces from the hop distance of the closest ingress identified so far. While the green bars indicate the total number of probes if the traces were to be stopped right after determining the ingress point of the AS, red bars indicate the additional probe packets generated for complete tracing. As we are looking for the same ingress from both ends of a subnet prefix, observing a different IP at the ingress hop would indicate the mismatch. On average, 58.2% of probes to ASes would have been eliminated if traces were stopped at the ingress point of the destination AS while overall 60.4% of all of the 238 745 538 probes would have been eliminated.

Figure 7a presents the number of edges that were discovered during the IPI phase. While the green bars indicate the total number of observed edges if the traces were to be stopped right after determining the ingress point of the AS, the red bars indicate the additional edges observed on the complete

paths. On average, 30.4% of edges would have been observed if traces were stopped at the ingress of the destination AS while overall 24.6% of all of the 742 440 edges would have been observed. Note that, the observed edges might belong to other ASes.

Figure 7b presents the number of IPs that were discovered during the IPI phase. Note that, IPs that belonged to the destination AS were added to the destination IP list. On average, 12.7% of IPs would have been observed if traces were stopped at the ingress point of the destination AS while overall 8.7% of all of the 462 973 IPs would have been observed. Note that, the observed IP might belong to other ASes.

B. Destination Network Tracing

In the following, we compare proposed InSub approach to the commonly utilized aggressive approach of tracing all destinations from all vantage points and tree based approach. Note that, we did not compare against the coordinated methods of [10] and [5] as their code were not available. Also, our method differs by being a decentralized approach.

Figure 8a presents the destination IPs that were probed during topology discovery based on the ingress to subnet relation. The total of 592 286 IPs are combined from the 258 955 from ANT Census [15] and 384 645 from IPs observed during the IPI phase of ingress to subnet discovery. Light blue part indicates IPs unique to ANT census, green part indicates IPs unique to IPI discovery phase, and red part indicates common IPs of both. Note that ANT Census data contain IPs from replies to an ICMP ping between June - August 2015 whereas IPI data contains IPs that generated ICMP Time Exceeded replies.

Figure 8b presents the number of vantage points used with each of the AS. Green bars indicate the number of vantage points utilized in measurement of each AS while additional red bars indicate the vantage points that were selected but did not complete probing. The red vantage points are due to PlanetLab monitors that either failed or became unreachable during the measurement. While AS6453 is the only AS probed by all of the assigned vantage points, AS3356 had the largest number of lost vantage points with 21 monitor failures.

Figure 8c presents the number of ingress points identified per AS. Higher number of ingresses allow for a better mapping of an AS as the ingresses are the real vantage into the AS.

Figure 9 presents the number of probes generated by each approach of *Full* (where all IPs are traced from all available vantages without any reduction), *All* (where traces start from ingress hop), *InSub* (where only one vantage traced subnets with respect to each ingress point), *Tree* (where each vantage traced all IPs in a tree-like fashion), and *InSub & Tree* (where tree is applied on top of InSub). Note that, all approaches started tracing of destination IPs from the identified hop distance of ingress to subnet with respect to the vantage point discovered during the IPI phase. We observe that in total, *Full*, *All*, *Tree*, *InSub* and *InSub & Tree* generates 2 192 856 642, 1 244 190 006, 579 337 950, 555 686 688 and 437 499 165 probes, respectively. Compared to extensive *Full*

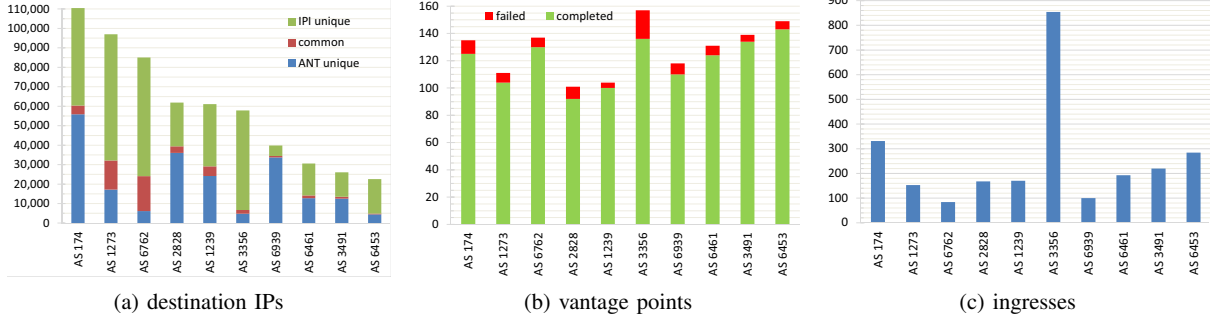


Fig. 8: Destination network tracing

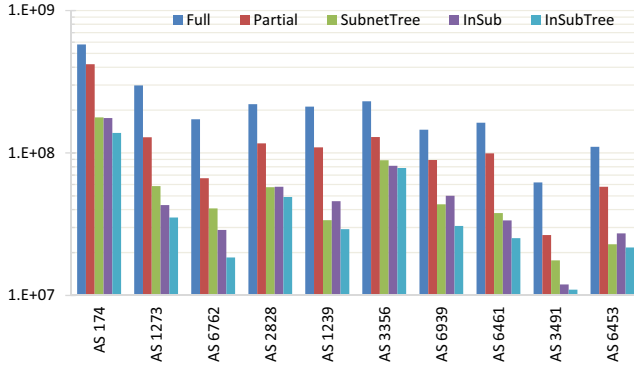


Fig. 9: Probes generated for network tracing (log-scale)

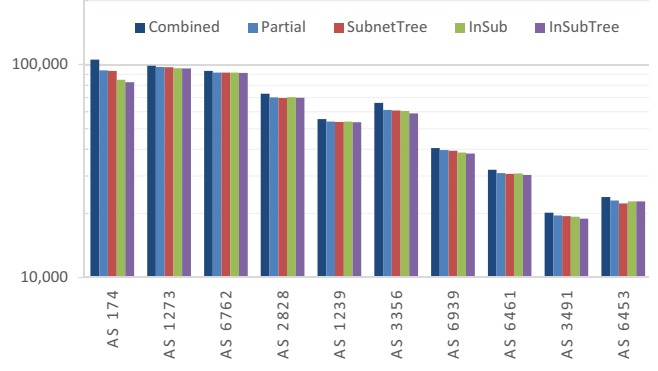


Fig. 10: Observed IPs per AS for each method (log scale)

approach, on average of all ASes, *All*, *Tree*, *InSub* and *InSub & Tree* approaches generate 46.7%, 74.3%, 75.6% and 81.0% fewer probes, respectively. Overall, *InSub* approach saves more probes than *Tree* approach while they can together further reduce probes. Considering sum of all probes, *All*, *Tree*, *InSub* and *InSub & Tree* approaches generate 43.3%, 73.6%, 74.7% and 80.0 fewer probes, respectively compared to *Full*.

Figure 10 presents the number of observed IP addresses with each approach. Combined indicates the union of all approaches. In total, *All*, *Tree*, *InSub* and *InSub & Tree* approach discovered 581 706, 578 145, 568 930 and 562 513, respectively, of all of the 608 667 combined IPs. Figure 11 presents the ratio of IPs discovered by each method compared to the combined union of all approaches. On average, *All*, *Tree*, *InSub* and *InSub & Tree* approach discovered 96.0%, 95.2%, 94.4% and 93.3% of each AS'es combined IPs, respectively. Considering sum of all IPs of each method, *All*, *Tree*, *InSub* and *InSub & Tree* approach discovered 95.6%, 95.0%, 93.5% and 92.4% of combined IPs, respectively.

Figure 12 presents the number of observed edges with each approach. Combined indicates the union of all approaches. In total, *All*, *Tree*, *InSub* and *InSub & Tree* approach discovered 2 842 622, 2 698 174, 2 581 213, and 2 476 526, respectively, of all of the 3 170 984 combined edges. Figure 13 presents the ratio of edges discovered by each method compared to the union of all approaches. On average, *All*, *Tree*, *InSub* and

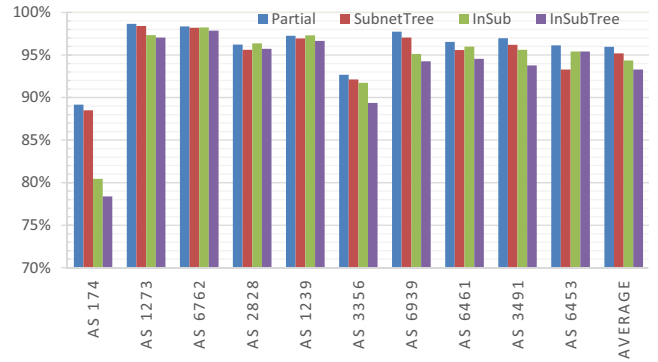


Fig. 11: Ratio of IPs discovered by each method

InSub & Tree approach discovered 89.2%, 83.2%, 83.9% and 79.9% of each AS'es edges, respectively. Considering sum of all IPs of each method, *All*, *Tree*, *InSub* and *InSub & Tree* approach discovered 89.6%, 85.1%, 81.4% and 78.1% of combined IPs, respectively.

While *InSub* misses 0.7% IPs compared to the *Tree* approach, it observes 0.5% more edges among them. As the tree-like paths of the *Tree* approach limits edge visibility, *InSub* can reveal more edges. On the other hand, *InSub* missed IPs as some of the failed vantage points led to failure of tracing of some IPs. Hence, such vantage point failures should be dynamically handled to address such losses.

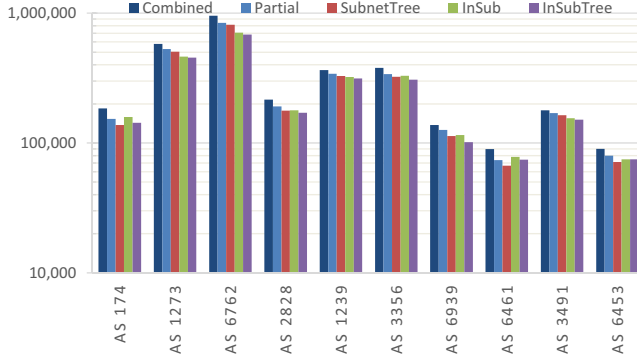


Fig. 12: Observed edges per AS for each method (log scale)

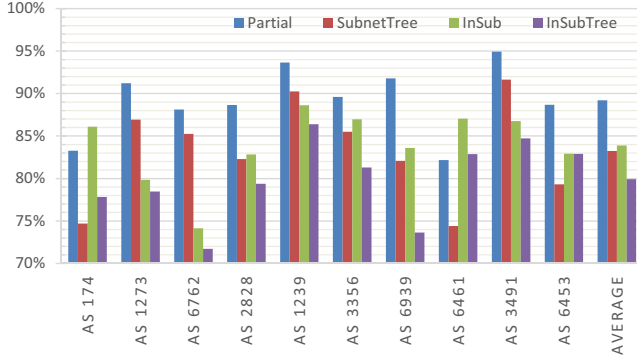


Fig. 13: Ratio of edges discovered by each method

C. Network Characteristics

In this subsection, we compare the network characteristics of collected topologies by each approach to see whether they produce similar graphs.

Figure 14 shows the CCDF of the degree distribution of the four approaches for each AS respectively. We observe that the degree distributions show similar patterns in all of the ASes.

Figure 15 presents the assortativity of each approach. Assortativity value is in the range of $[-1, 1]$. A value of 1 indicates assortative behavior, i.e., high degree nodes are linked to other high degree nodes, a value of -1 indicates dissortative behavior, i.e., high degree nodes are linked to low degree nodes, and a value of 0 indicates non-assortative behavior. All networks are either slightly dissortative (e.g., AS174 and AS2828) or dissortative. We observe that almost all approaches produce graphs with similar assortativity values. Only in AS3356 there is difference of more than 0.1 but note that this AS had the most VP failures during tracing.

Figure 16 presents the clustering coefficient of each approach. Clustering value is in the range of $[0, 1]$ and is calculated as the number of triangles divided by the number of triplets in the graph. A value of 0 indicates lack of triangles in the graph while a value of 1 indicates a clique topology. We observe that all graphs have a similar clustering coefficient.

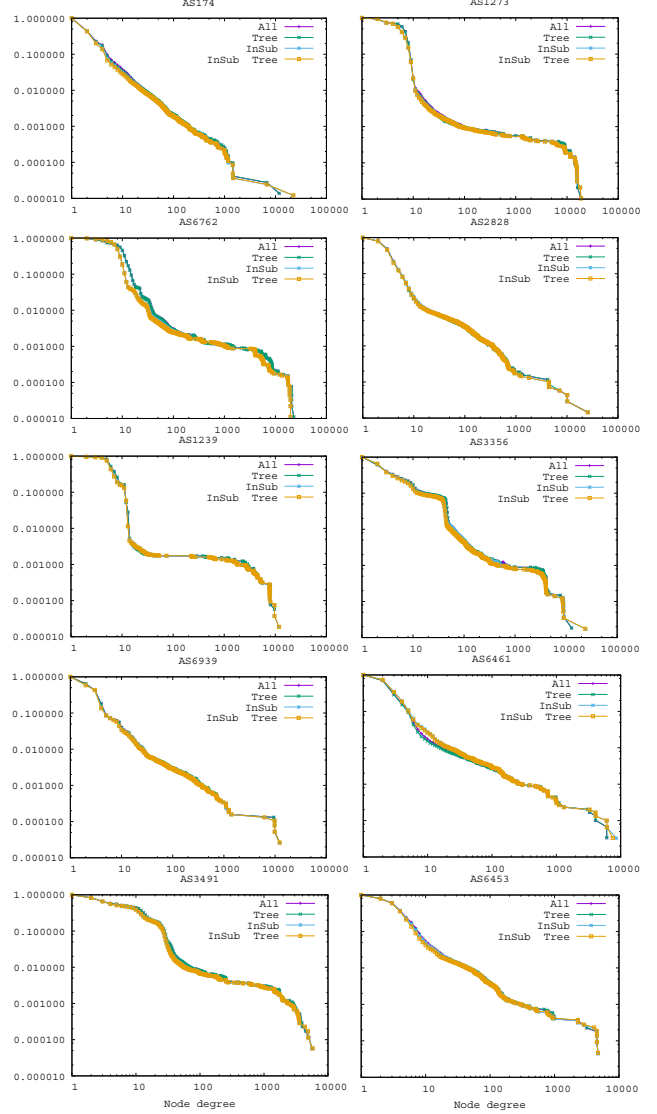


Fig. 14: CCDF of degree distribution of the resulting graphs (log-log scale)

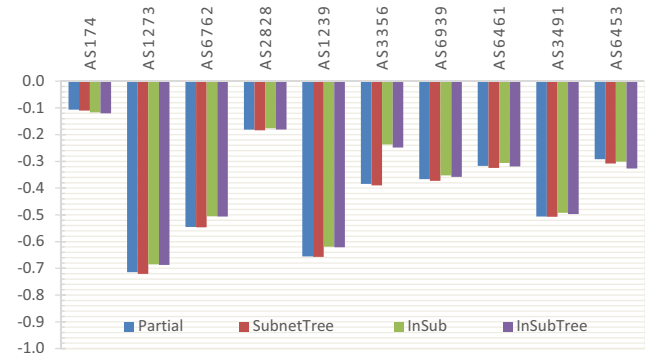


Fig. 15: Degree assortativity of the graphs

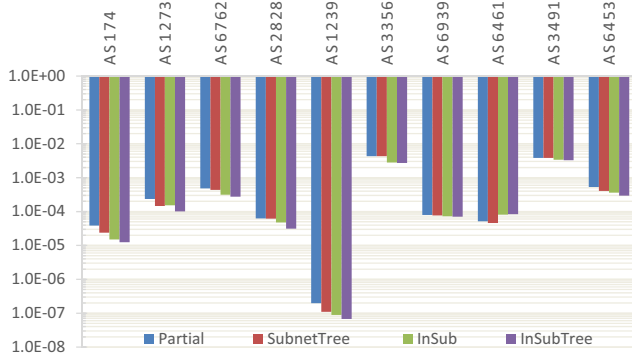


Fig. 16: Clustering coefficient of measured topologies (log-scale)

V. CONCLUSIONS

In this paper, we present a new approach for minimizing redundant probes while mapping a destination network topology. *InSub* approach is based on ingress identification of subnets along with vantage point allocation to optimize probing. *InSub* first identifies ingress points of announced subnet prefixes by tracing subnet end-points. During this process, *InSub* recursively splits subnet prefixes that enter the destination AS through different ingress points. Then, *InSub* probes each subnet from one vantage point per identified ingresses of the AS. This considerably reduces the number of generated probes without significant loss in the observed topological information.

As we did not account for loss of vantage points during the measurements, we lost some of the information with the *InSub* approach. As *InSub* traces each ingress to subnet from one vantage point, losing a vantage point typically means that visibility is lost. Hence, in the future, we will develop a dynamic system that could replace the lost vantage points with respect to the ingresses. Additionally, we realized that our vantage points were able to capture only a small fraction of the peering relations of ASes reported by CAIDA. This indicates, we were not able to probe through most of the ingress points of ASes. In the future, we will look for a larger number of vantage points to have a better visibility of a target AS.

ACKNOWLEDGMENT

This work is supported by the NSF grant CNS-1321164.

REFERENCES

- [1] B. Li, J. Springer, G. Bebis, and M. H. Gunes, "A survey of network flow applications," *Journal of Network and Computer Applications*, vol. 36, no. 2, pp. 567 – 581, 2013.
- [2] H. Kardes, M. Gunes, and T. Oz, "Cheleby: A subnet-level Internet topology mapping system," in *IEEE COMSNETS*, 2012, pp. 1–10.
- [3] E. Arslan, M. Yuksel, and M. H. Gunes, "Training network administrators in a game-like environment," *JNCA*, vol. 53, pp. 14–23, 2015.
- [4] Y. Shavitt and U. Weinsberg, "Quantifying the importance of vantage points distribution in internet topology measurements," in *IEEE INFOCOM*, 2009.
- [5] B. Donnet, P. Raoult, T. Friedman, and M. Crovella, "Efficient algorithms for large-scale topology discovery," in *ACM SIGMETRICS*, Jun 2005, pp. 327–338.
- [6] B. Cheswick, H. Burch, and S. Branigan, "Mapping and visualizing the Internet," in *USENIX Annual Technical Conference*, 2000, pp. 1–12.
- [7] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with rocketfuel," in *ACM SIGCOMM CCR*, vol. 32, 2002, pp. 133–145.
- [8] "University of Oregon Route Views Project," <http://www.routeviews.org/>.
- [9] N. T. Spring, D. Wetherall, and T. E. Anderson, "Scriptroute: A public Internet measurement facility," in *USENIX USITS*, 2003.
- [10] R. Beverly, A. Berger, and G. G. Xie, "Primitives for active Internet topology mapping: Toward high-frequency characterization," in *ACM Internet Measurement Conference*, 2010, pp. 165–171.
- [11] "Ripe ncc. "Announced prefixes"," <https://stat.ripe.net/>.
- [12] G. Balra, R. Beverly, and G. G. Xie, "Ingress point spreading: A new primitive for adaptive active network mapping," in *Passive and Active Measurement*, 2014, pp. 56–66.
- [13] H. Kardes, M. H. Gunes and K. Sarac, "Graph based induction of unresponsive routers in Internet topologies", *Computer Networks*, 2015, vol. 81, pp. 178–200.
- [14] "Planetlab," <http://www.planet-lab.org/>.
- [15] "Ant censuses of the Internet address space," <https://ant.isi.edu/address/>.
- [16] B. Augustin, X. Cuvelier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira, "Avoiding traceroute anomalies with paris traceroute," in *ACM Internet Measurement Conference*, 2006, pp. 153–158.
- [17] C. Pelsler, L. Cittadini, S. Vissicchio, and R. Bush, "From Paris to Tokyo: On the suitability of ping to measure latency," in *ACM Internet Measurement Conference*, 2013, pp. 427–432.
- [18] M. H. Gunes and K. Sarac, "Analyzing router responsiveness to active measurement probes", *Lecture Notes in Computer Science*, 2009, vol. 5448, pp. 23–32.
- [19] "Caida AS ranking," <http://as-rank.caida.org/>.