Scaling Reverse Traceroute: An Evaluation of Vantage-Point Selection Strategies

Ethan Katz-Bassett, Brian Goodchild, Rohan Mehta, and Alexander Stein

Columbia University

Abstract.

Reverse Traceroute is a tool for measuring IP-level hops from a destination D back to a source S. The primary mechanism used to implement the measurement is the Record Route (RR) IP Option, which records the first nine L3 hops on the round-trip path S $\rightarrow D \rightarrow S$. When S is greater than eight hops away from D, this system relies on sending RR-enabled pings to D from in-range vantage points (VPs) spoofing S as their source address. Overuse of these probes risks prompting the adoption of filtering policies by network administrators, and triggering rate-limiters which may produce inaccurate network views and thus inaccurate results.

In view of these risks, we investigate techniques that allow us to scale the Reverse Traceroute system by minimizing the number RR-pings that must be issued before one is successful, and by maximizing the number of reverse-hops appearing in the RR-header of successful pings. In particular, our focus is on optimizing the VP rankings used in the out-of-range scenario described above. Prior work[6] - with offline data from daily probes - creates these rankings in two ways: a greedy approach which ranks VPs by the magnitude of their coverage over previous measurements into the target destination network, and a restrictive approach using only the global top K[1] VPs in terms of number of destinations in-range. For convenience, we term these two algorithms "destination cover"and "top-K", respectively. The primary contribution of our work is the addition of a novel VP-ranking algorithm, "ingress scoring" (IS). IS ranks each VP based primarily on hop-distance to a destination network's ingresses, viewing as redundant additional-but-equidistant VPs converging to the same ingress.

Our analysis compares the efficacy of the ranking algorithms (DC, TK, IS); considering multiple definitions of "ingress" relative to the BGP-routeable prefix containing a destination address. The preliminary iteration of this approach - termed "ingress cover" (IC) - employed greedy set-covering to create "logical ingresses." While initial results did show IC selecting VPs much closer to destinations on average than either of the other algorithms, IC suffered an increase in the number of unreachable destinations. This followup work eliminates that shortcoming by refining how ingresses are defined and how VPs are ranked relative to those ingresses, while still relying on the same core concept: Internet routing is destination-based, and so paths from disparate VPs towards the same destination will tend to converge as they near their target. New results demonstrate that IS uncovers over 50% more reverse-hops and reaches slightly (2%) more destinations than both DC and TK.

Keywords: Reverse Traceroute · Internet Measurement · Internet Routing · Ranking Algorithms · Scalability.

[1] We use $K = 10$ in our work.

1 Introduction

Insight into routing on the reverse path from an uncontrolled destination back to a source under your control has long been sought after for diagnosing issues such as inflated latency due to circuitous routing on the reverse path[9]. The prevalence of such paths on the Internet has led some network operators to build out extensive backbones in order to ensure more direct routes to and from customers[10]. While it is often possible for network operators to inject measurements into client-downloaded content through tools such as JavaScript[3], there remain cases in which it is not possible to request that a destination issue a measurement towards a source. In these cases, a traditional traceroute tool is useless, as it requires a source to vary TTL values, then record the responses from routers along a path. Reverse Traceroute[8] is a tool for measuring IP-level reverse paths from an uncontrolled destination back to an arbitrary source. It utilizes the IPv4 Record Route Option to record hops on the reverse path.

1.1 Background

In order to give scope to the problem of VP selection, we begin with a brief overview of the way in which Reverse Traceroute[8] uses the IP Record Route to gather reverse path information. This system is currently operational for use in reverse path measurement at revtr.neu.ccs.edu[1]. Reverse Traceroute enumerates hops along destination-to-source ($D \rightarrow S$) paths and "stitches them together" through repeated active RR-enabled ICMP ping probes (RR-pings). The Record Route(RR) IP option requests that routers record their IP addresses in the packet header before forwarding the packet. Up to 9 IP addresses can be saved to the RR-header in this fashion; after which point the RR-header is forwarded without further modification.

The key insight underlying this technique is that ICMP echo replies copy original packet headers into their responses, and so RR-ping responses also have RR enabled. Should fewer than 8 IP addresses be recorded on the way from $S \rightarrow D$, then there are "slots" available to be filled first by D and then by routers on the return journey. Once at least one router IP-address R on the reverse path is discovered, this process is repeated - with each new R as the destination - until a full reverse-path is completed.

A given source is seldom within the required 8-hop range of a destination. Reverse Traceroute employs a set of globally distributed Vantage Points (VPs) to circumvent this limitation in the following way. As the path from S to D is desired, a VP V believed to be within range of D is selected from which to issue an $V \rightarrow D$ RR-ping spoofing S's IP-address as the source. Because the source header-field is listed as S, the ICMP echo reply will be directed towards S instead of V. These spoofed probes bootstrap construction of reverse paths until they are within range of S, or until they intersect with an atlas of known reverse paths. The method by which VPs are selected for this purpose has an outsized impact on the overall system performance, and is the subject of this paper.

1.2 Challenges

Record Route suffers from two fundamental limitations that hamper the scalability and performance of a Reverse Traceroute system. First, since RR-pings can record at most 9 IP-addresses, the system relies heavily on repeated use of the distributed VPs. The measurement platforms that host these VPs are reluctant to allow spoofing, which can be abused if not handled properly. This effectively restricts the set of VPs that can be incorporated into a Reverse Traceroute system. Our current system is implemented on a set of roughly 100 M-Lab and Planetlab nodes[2].

Second, packets with Record Route specified in the IP header are - like other IP Options packets - often considered undesirable to route due to the extra processing required at routers along a path[7]. Thus network operators may configure their routers to drop Record Route packets sent in excess of very conservative rates, such as 10-20 packets-per-second[6]. We refer to these routers as "rate limiters," and the act of sending in excess of the configured rate as "triggering." This severe rate-limiting can hamper efforts to issue large amounts of concurrent measurements, a requirement for a scalable Reverse Traceroute system.

These two limitations present a challenge. It may be possible to "spread out" measurements to new destinations in such a way to prevent overloading destination networks and triggering destination-proximate rate-limiters. However, since there are few VPs relative to destinations, it is impossible to "spread out" measurements across VPs. This means that vantage-point-proximate rate limiters can be easily triggered by a system making measurements at the scale of other popular measurement platforms. Such rate limiters have been shown to drastically decrease the number of successful measurements on our system[6].

To prevent triggering source-proximate rate limiters, we require prudence in issuing new measurements – sending only probes from VPs that we expect will be within range of a targeted destination. Such prudence comes with challenges, however. First, any selection methodology that restricts the set of VPs from which to issue new measurements will be at risk of false negatives – incorrectly reporting that no vantage point is within range. Second, though it may be possible to find a vantage point that is "at least" in range, we would like to find one that is of minimal distance to the destination. This allows us to record more hops on the reverse path with fewer repeated probes.

2 Design

When the source of a Reverse Traceroute measurement is not within range of the destination, we must choose a Vantage Point from which to attempt measurements. Due to the fact that IP options are strictly rate-limited, we want to choose a VP in such a way that

1. a minimal number of RR-pings are required before finding a VP in range, and

2. a maximal number of hops can be recorded on the reverse path, potentially precluding the need for excessive measurements to "stitch together" the reverse path.

We compare three techniques developed to achieve these goals. All three techniques utilize past measurements issued through the Reverse Traceroute system to augment the intersection atlas, informing the decision of which VP to select. For the purpose of this work, we refer to a VP as *RR-reachable* to a destination if, when sending an RR-ping to the destination from that VP, the destination IP appears in the packet header of the response. This means that the RR-ping arrived at the destination with enough free space in the header to record at least one IP address.

2.1 Top-K Ranking

Previous work has shown that a small subset of our VPs is sufficient to reach the majority of destinations on the internet within 9 RR hops[3]. As a baseline, we implement a Vantage Point selection technique that restricts the number of VPs to only the top K in order of ranking by distance to sets of destinations. First, we construct a set, S, of all previously RR-reachable destinations seen over the course of some time window (a day to a week) from any VP in our measurement platform. Next, we rank VPs using a greedy set cover approach. Over several iterations, the VP chosen for the next place in the ranking is the one that is RR-reachable with respect to the greatest number of remaining destinations in S. After each round, the destinations RR-reachable from the chosen VP are removed from S. When the full ranking of VPs is complete, we choose our top K.

2.2 Destination-Cover Ranking

This approach is similar to the selection technique deployed by the original Reverse Traceroute system[2]. The method works as follows. When a new destination, D, is to be measured towards, and no previous measurements to D, have been recorded by the system, we first map D to its "destination network." This could be the /24 prefix, BGP-routable prefix, or the ASN number to which D belongs. Next, we consider all RR-reachable destinations from the same destination network as D[2]. Similarly to the Top K approach, we use greedy set cover, but this time restricting the set S to only these destinations. Vantage Points are chosen in the order they are selected in the set cover algorithm.

2.3 Ingress Ranking

Our novel approach takes advantage of the fact that routes tend to converge as the approach a fixed destination. By finding these points of convergence, or

[2] We choose *BGP-routable prefix* as our definition of *destination network*. As internet routing is destination-based, we believe this to be the most representative possible definition of those discussed.

"ingresses," we can issue measurements from the Vantage Points that are closest to each ingress and preclude measurement from all other VPs. This allows us to not only rank VPs based on their relative distance to destination networks, but also gives conditions under which it should be safe to cease attempting future measurements.

The technique works as follows. For a new destination, D, to which no previous measurements have been made, we first map D to its BGP-routable prefix; next, we use past RR measurements to map VPs to their "ingresses". How "ingress" is defined can be varied over (1) which type of network (ASN, BGP, /24) is being entered, and (2) its position relative to that network (last hop before entering, first hop after entering, etc.). The investigation of other ingress definitions is an open problem left to future work.

Once all VPs are mapped to their ingresses, we can rank the VPs that share the same ingress based on their distance to the ingress, in terms of the number of hops recorded in the packet header before the ingress itself. In so doing, we can issue measurements only from the closest VP to each ingress. Since routing is destination-based, we can expect RR-pings that traverse the same ingress to follow the same route after reaching the ingress. This means that we need only measure from the closest VP to each ingress.

However, this form of mapping is not always consistent. Many times, a single vantage point can be seen to traverse multiple ingresses on the way to the same destination network. This could be due to coarse definitions of "destination network" in which the actual destinations are not very similarly routed, due to load balancers that send packets along different routes depending on network conditions, or even due to the fact that some routers can record multiple different IP addresses in the packet header across repeated measurements (known as aliases).

Ingress Cover The subject of our previous work, an initial approach to dealing with this problem maps each Vantage Point to the set of IP addresses determined to be ingresses into D's prefix. On completion, sets of ingresses are aggregated with sufficient overlap into single functional ingresses. For example, if vantage point V1 is seen to traverse ingress IP addresses i1, i2, i3, and Vantage Point V2 traverses ingresses i2, i3, i4 then we will refer to the set i1, i2, i3, i4 as one logical ingress, I, and rank both vantage points V1 and V2 with respect to I. The amount of overlap between sets of ingresses required before they are merged into logical ingress sets is a configurable parameter, and has been tested at 0.75.

Our evaluation of *Ingress Cover* reveals significant drawbacks to the approach. Aggregating ingresses into logical groupings is a destructive process that can discard valuable distinctions. For example, two ingresses I_1 and I_2 may each be optimal for two different unknown sub-networks S_1 and S_2, respectively. Suppose $|V_A \longrightarrow I_1| = 2$, $|V_A \longrightarrow I_2| = 6$, $|V_B \longrightarrow I_1| = 7$, and $|V_B \longrightarrow I_2| = 3$. Clearly we want to pick V_A for destinations in S_1 and V_B for destinations in S_2; but the former approach will throw V_B away as it is further from the logical group.

Furthermore, it is not necessarily reasonable to treat each of potentially several network-ingresses traversed by past measurements from a single VP as equally likely to be used again in the future. Due for example to temporary load-balancing conditions, some VP normally very far from the target network may have recorded a small number of short-distance measurements through an ingress unlikely to be used when these conditions subside.

Ingress Scoring To address the above shortcomings, we alternatively propose *Ingress Scoring*. This approach chooses not to aggregate ingresses, instead simply giving a distance-based total ordering over all VPs with past measurements into the target network for each ingress. This produces a much larger set of ingresses to consider than the previous method, and so - to avoid the potential for issuing many probes before a "good" ingress is found - it is necessary to intelligently order the ingresses themselves.

Towards this end we first introduce a novel heuristic called *popularity score* that measures the frequency with which a VP used its most popular ingress when considering all its measurements into some destination BGP-prefix:

$$pscore = \frac{\max_{\forall Ingr \in meas(V,Prefix)} \sum_{m=meas(V,Prefix)} 1 * (Ingr \in m)}{|meas(V, Prefix)|} \quad (1)$$

We did a brief analysis of the distribution of popularity scores over all $\{VP, Prefix\}$ tuples, varying ingress definitions and comparing network sizes, the results of which are shown in Figure 2. A high popularity score lends confidence to the possibility that a previously used ingress will be used again in the future. It does not, however, imply that using said stable ingress will return a VP which is optimally close to it when compared with all other network-ingresses.

Using the weighted average of each ingress's normalized popularity score and normalized minimum VP-distance allows us to construct a robust ranking that satisfies both stated design goals:

$$iscore_{Ingr} = \frac{w_p * norm(pscore_{Ingr}) + w_d * norm(\min_{vp \in Ingr} dist)}{2} \quad (2)$$

The popularity and min-dist weights have been experimentally shown to best optimize design goals when tuned roughly to $w_d \approx 3w_p$.[3]

[3] It has been suggested that perhaps a better heuristic for *iscore* would be the *expected distance*:

$$E[D_{Ingr}] = \sum_{d=1}^{9} \frac{|vp \rightarrow Ingr : dist == d|}{|\forall vp \rightarrow Ingr|} * d \quad (3)$$

3 Evaluation

In the first iteration of this work (producing *Ingress Cover*), there were several flaws in the evaluation methodology that decreased data quality and obfuscated the ability to meaningfully compare ranking approaches. Some of these have already been addressed and are discussed below; others remain part of ongoing work to be reviewed in the next section.

3.1 Case Studies

In order to discover which aspects of *Ingress-Cover* (or our evaluation methodology) were problematic, we performed dozens of case-studies examining which algorithms performed best/worst for specific destinations and why. Though each case-study required independent critical examination, all took the same general form. First, a test destination is identified and the emulation results for each ranking algorithm for that destination are gathered for comparison.

Dirty Data. Several initial case studies were difficult to extract meaning from because of the data collection issues discussed in section 3.2 and the inconsistency present in our original emulation strategy w.r.t. the selection of testing and training data. For example, the initial set of IP addresses to which measurements were later issued was not filtered, and so "bad" destinations like those in the private address-space produced results that were red herrings from which no conclusions about algorithm efficacy could be drawn. These types of investigations were critical since failing to find & fix such problems would cast doubt on the relevance of our aggregate results.

Rate Limiting. Some VPs had to be removed from consideration after initial case studies indicated their suffering from source-proximate rate-limiting. A very large AS - Korea Telecom (ASN 4766) - contains many destinations for which our case studies raised red flags. Of particular interest is the fact that while most VPs had about 65,000 RR-responsive measurements into this AS, 10 VPs had less than half as many, with the worst case being the VP mlab1.bcn01.measurement-lab.org which had only a single such measurement.

BCP 186. This case-studying process did not just uncover the shortcomings mentioned in Section 2.3; it also lead to the discovery of hard evidence for a suspicion we earlier theorized in [6]. RFC-7126[5] discusses best practices regarding the filtering of IPv4 packets containing IP options; specifically advising

> *"Packets with this IP option are forwarded as if they did not contain this IP option."*

The authors of [6] in fact use this recommendation as part of their motivation for reassessing the validity of the RR-option. Surprisingly, our recent case studies

have repeatedly demonstrated instances in which ASes appear in a traceroute to D, but not in an RR-ping to D - evidence of forwarding RR-headers without stamping. This often has the impact of "short-circuiting" the ingress ranking algorithms. For example, in Figure 1 there is one VP, planetlab1.dtc.umn.edu, which appears from the IP-addresses recorded in its RR-header to bypass the most popular ingresses. An ICMP traceroute issued towards this destination from planetlab1.dtc.umn.edu, however, contradicts this thought: there are in fact 10 hops from start to finish. The degree to which this policy has been adopted and effects the results of our evaluation is the subject of ongoing work.

Logical Ingresses. Of primary concern when assessing *Ingress-Cover* results was the usefulness of *logical ingresses*. *Logical* or *Functional* ingresses are formed by mapping each VP to the set of IP addresses that are determined to be ingresses into the target destination's network. Once this mapping is complete, sets of ingresses are aggregated with sufficient overlap into single functional ingresses. The output of this ranking algorithm is a list of logical ingresses and a single, closest VP per logical ingress. Our initial rounds of emulation found this strategy to fall far short of both top-k and destination-cover in terms of the number of test-destinations successfully reached via attempted VPs from their corresponding rankings. Even filtering out rate-limited VPs and "bad" destinations did not improve this rate of coverage. We thought that perhaps this issue was due to only a single VP being returned per ingress, but coverage hardly improved even for a full emulation rerun in which a total distance-ordering over all available VPs per ingress was used. An important takeaways from this experience is that the aggregation step required to build a logical ingress lumps together routers that are frequently traversed with those that are infrequently traversed, as well as making the questionable assumption that VPs close to one router in the grouping are close to all routers in the grouping.

Physical Ingresses. Persistent failure in *Logical Ingress* coverage prompted us to cast a wider net and see if there existed *some* concept of ingress that reliably produced the expected path convergence characteristics. We theorized that "ingress" might be better defined by (1) using actual routers seen in RR-headers as ingresses (*Physical*), and (2) varying the type of network (BGP-prefix, ASN, /24) used to place the ingress relative to the fixed destination BGP-prefix containing the destination in question. Before scaling this hunch out to hundreds of lines of code in the full emulation, we first verified it on several hand-picked test destinations.

3.2 Dataset

We randomly picked 13,000 historically ping-responsive /24 prefixes from a recent ANT ISI hitlist. For each of these /24 prefixes, we sampled 50 IP addresses randomly; resulting in a total of approximately 646,000 destinations. From our

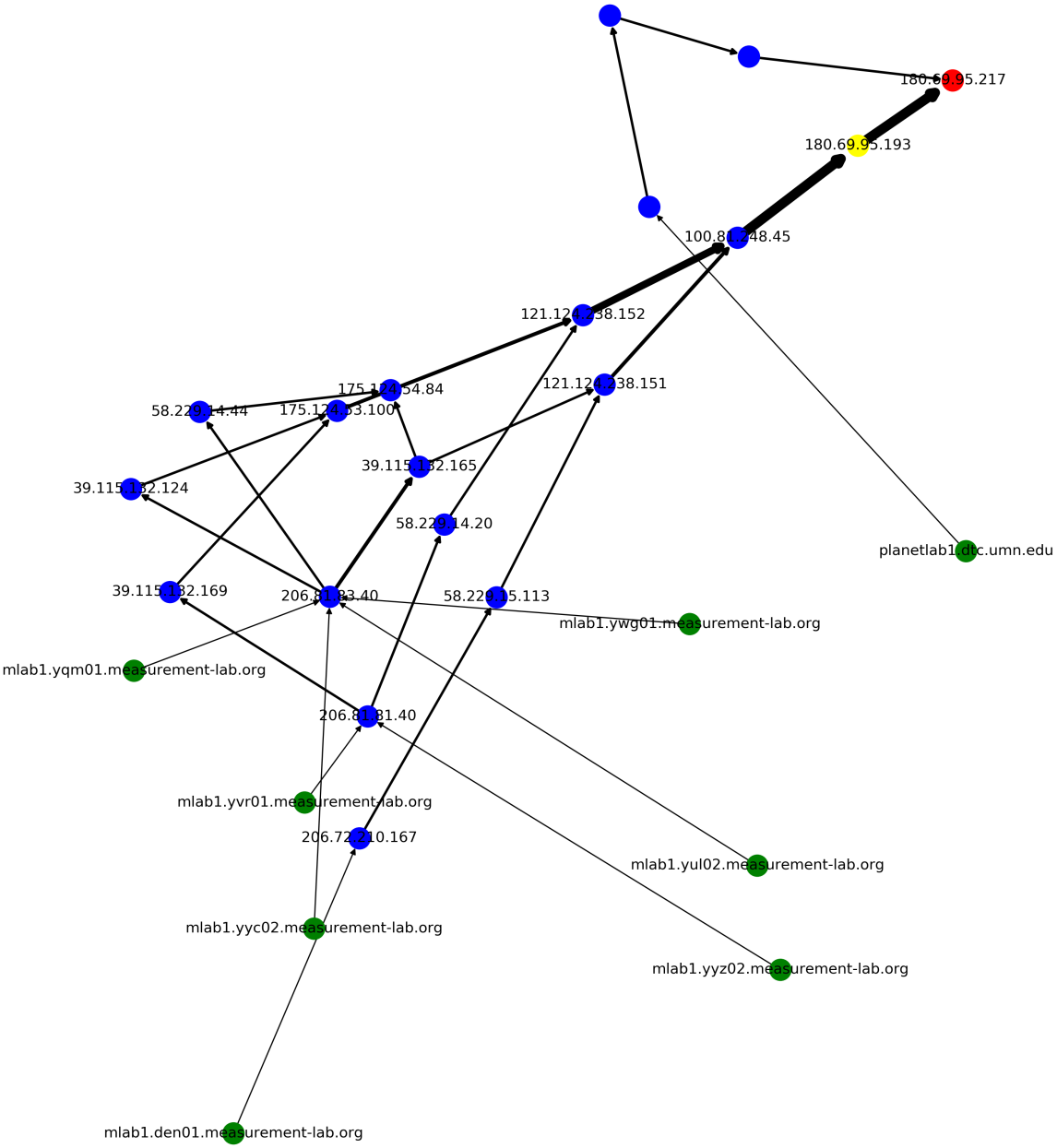


Fig. 1: Sample case study illustrating vantage points converging towards common ingresses as they near the destination. Line thickness indicates the relative fraction of measurements towards the destination which traversed this edge. The color coding is as follows: Green → VP , Red → Destination, Dark-Blue → Router. If a router is in the same /24 as the destination, it is colored yellow; same BGP-prefix gets cyan, and same ASN gets orange.

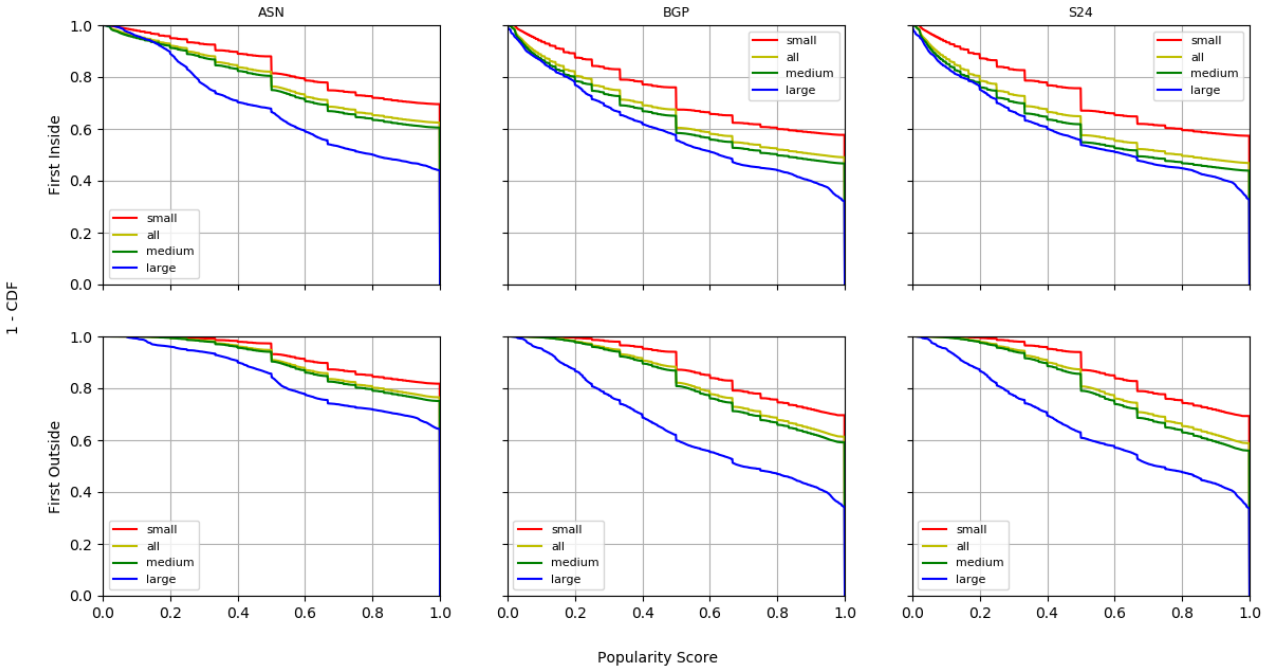


Fig. 2: CCDF of popularity scores over every $\{VP, Prefix\}$ tuple. One plot for each distinct definition of ingress. The tuples are further categorized by the relative sizes of their prefix. *Small* prefixes are those with at most $1,000$ IP addresses, *Medium* with between $1,000$ and $1,000,000$ IPs, and *Large* with at least $1,000,000$ and up to $1,000,000,000$ IPs.

set of 100 Vantage Points (96 M-Lab nodes and 4 PlanetLab nodes[4]), we issued one RR-enabled ping (in shuffled order) to each of the 500k destinations. That gave us 50 million RR-enabled measurements; each measurement containing the IP addresses of 9 hops as traversed by the ping packet. These probes were sent and collected over a period of 3 days in mid 2018, at which time a snapshot of the RouteViews BGP database was also collected to ensure valid $IP \rightarrow Prefix$ mappings during our analysis.

3.3 Training and Emulation

We define three notions of network, as described above: /24 prefix, BGP-routable prefix, and ASNs. Our original work did an 80%-20% split of measurement data *for each network definition* into training and testing sets, respectively. The key difference in this latest work is that only BGP-routable prefixes are considered for creating test/train sets. Results from the original study were difficult to compare because destination IPs could fall into both testing and training sets across different network definitions; trying to parse out what causes one ranking-algorithm to outperform another without complete information on a given test-case is quite difficult.

[4] PlanetLab vantage points are being phased out of service; our data collection pipeline just so happened to uncover 4 available PlanetLab nodes during a health-check.

We compare *Ingress Scoring* with each other algorithm, and with "optimal VP-selection" i.e. against one of the VPs found to be the minimum possible distance from the test-destination given full view of the test data set. The training set had about 508,000 measurements, and the testing set about 138,000 measurements, both from each of 87 VPs (down from 100 after several were removed from consideration after being identified as suffering from source-proximate rate limiting). The metrics considered for each ranking method (and across differing definitions of ingress) are *coverage*, *distance*, and *pings*. "Coverage" is defined as the fraction of test-destinations RR-reachable from optimal VPs which is also RR-reachable from VPs chosen by the given ranking algorithm. "Distance" is the average difference between the hop-distance from the optimal VP to the destination and that of the chosen VP. "Pings" represents the average number of attempts (emulated RR-ping probes) that were made by the algorithm before a VP was found which was RR-reachable to the destination.

Collecting these metrics is fairly straightforward for *Top-K* and *Destination-Cover* because these methods produce the same results independent of how "ingress" is defined. For *Ingress Scoring*, on the other hand, it is not so simple. Ingress rankings have to be created from the training data for six definitions of ingress, and the data-structure storing those rankings is complex (max-heap of ingresses keyed by *iscores*, each of which contains a min-heap of VPs keyed by distance).

3.4 Results

The hop-distributions of each ranking method - broken down by type of destination network - are found in figure 3, and the coverage of reachable destinations for each is recorded in Table 1. Just as with the original study evaluating *Ingress Cover*, our new results illustrate that the *Ingress Scoring* method (yellow) comes very close to matching the optimal hop-distribution (red) for the best definitions of ingress. However, unlike the earlier work - in which there was a coverage penalty suffered by ingress-cover - ingress-scoring achieves slightly *better* coverage even than top-k. The closeness of ingress-scoring to optimal is quantified in Table 2, which shows that - on average - it strays far less from the optimal number of hops to a network than do destination covering and top-k approaches.

In addition, we tallied the number of RR-pings issued during emulation before the destination address was found as one of the 9-hops recorded by the RR-option from the chosen VP. If a ranking method did not return any VPs for a given destination network, then that destination was not counted towards the average. These results are summarized in Table 3: ingress-scoring roughly matches destination cover in number of pings. As *Ingress-Scoring* uses a similar number of probes to destination cover, and uncovers 2x more reverse-path hops than Top-K, our algorithm seems a promising step towards moving the reverse traceroute system towards internet scale. This suggestion is made stronger still considering the algorithm's near-optimal hop distribution as shown in figure 3.

	Top-10	Destination-Cover	Ingress-Scoring (First Inside)	Ingress-Scoring (First Outside)
BGP	99%	96%	100%	100%
ASN	-	-	100%	97%
S24	-	-	100%	100%

Table 1: Percentage of the number of destinations which were reachable by optimal-VP also reachable by the VP selected for each ranking algorithm. There were $138,000$ destinations tested during emulation.

	Top-10	Destination-Cover	Ingress-Scoring (First Inside)	Ingress-Scoring (First Outside)
BGP	0.88119	1.41167	0.34110	0.38699
ASN	-	-	0.88788	1.15636
S24	-	-	0.24360	0.28392

Table 2: Average difference between distance (in hops) from $VP_{opt} \rightarrow D$ and distance from $VP_{chosen} \rightarrow D$ for each ranking algorithm.

	Top-10	Destination-Cover	Ingress-Scoring (First Inside)	Ingress-Scoring (First Outside)
BGP	1.36729	1.07033	1.41239	1.34117
ASN	-	-	1.92796	2.73682
S24	-	-	1.31043	1.25612

Table 3: Average number of RR-probes sent before an RR-reachable measurement was achieved.

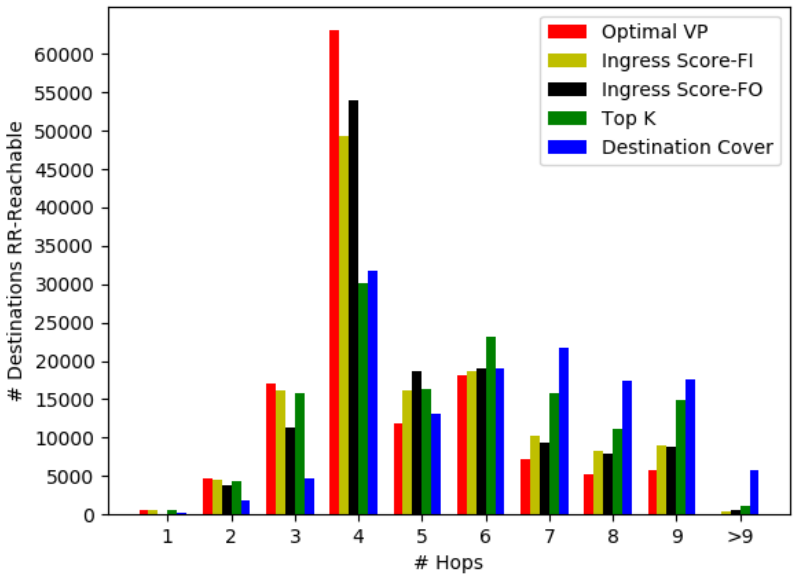
4 Discussion

Given the large upside and nearly non-existent downside observable from the collected metrics on *Ingress-Scoring*, we are nearly decided in pushing it into the production Reverse Traceroute system. There are however several loose-ends that must be tied up before this can happen.

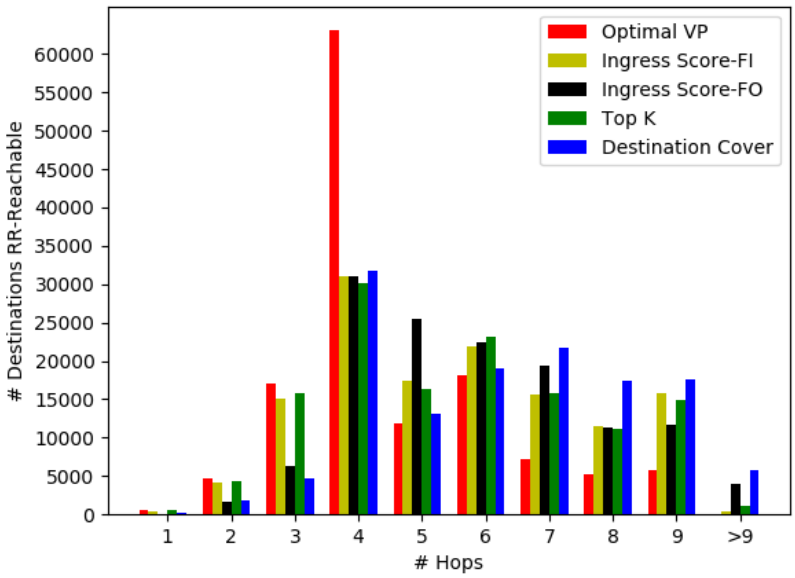
4.1 Ingress-Scoring: Open Issues

Ranking Runtime. As was briefly mentioned in the previous section, there is great complexity involved in emulating *Ingress Scoring* - so much so that it is often the computational bottleneck, taking more than 30 hours to complete. While this poor performance would certainly be an issue in an online Reverse Traceroute system that expects freshly updated results every day, it is nothing that cannot be fixed with some smart code refactoring.

Data Scarcity. Of greater concern is the amount of training data required to build out an ingress-scored ranking (or *any* ranking, for that matter). Especially vulnerable is the usefulness of our ingress-popularity heuristic in the face of

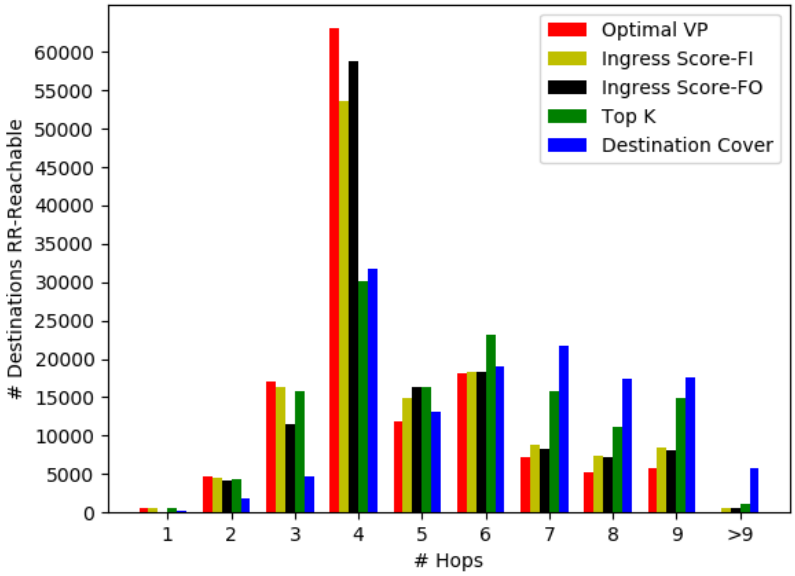


(a) ingress-network = BGP-prefix



(b) ingress-network = ASN

Fig. 3: Distribution of number of hops traveled from optimal/ranked VPs to all destinations.



(c) ingress-network = S24

Fig. 3: (cont.) Distribution of number of hops traveled from optimal/ranked VPs to all destinations.

sparse numbers of available measurements into most BGP-prefixes - which is the common case in the online system. Even in an online scenario in which no more than one measurement per prefix per VP per day is collected, we expect that *Ingress-Scoring* will continue to enjoy a high amount of coverage since the most popular ingresses appear from figure 2 to be quite stable. Investigating the degree to which this problem impacts online results is currently our highest priority item, as it actively blocks the system's promotion to a wider base of users.

Forwarding without Stamping. Discovering that some ASes have applied the BCP-186 recommendation to forward without stamping raises at least one potential issue: if enough networks adopt the policy, how representative can any stamps seen recorded in the RR-header really be of the actual round-trip path between S and D? Fortunately, our case studies have also shown that in instances of forwarding without stamping, the policy appears only to be applied in the forward direction, allowing for *even more* reverse-path IP addresses to fill the header than would otherwise be possible. After further investigation, we found this observation to be consistent with the disproportionate 4-hop mode visible in Figure 3. Indeed, two VPs ended up being responsible for over 95% of cases in which the optimal-VP for a test-destination was 4-hops away. Unsurprisingly, the mode shifts to 6-hops if these two VPs are excluded from consideration.

Smarter Data Collection. The way data was collected for our initial offline study had several drawbacks which are now being addressed in our ongoing measurement collection. Use of an ISI address *hit-list* to select /24 prefixes and then probe random addresses within that /24 carries with it the implicit (and

false) assumption that the existence of one ping-responsive IP in a /24 implies ping-responsiveness of all IPs in that network. In several instances we were unable to complete case-studies due to the inconsistency with which many VPs received/did not receive ICMP-echo-replies across destinations from the same /24. To address this problem our followup collection is using the ISI address *survey*, in which *all* IP-addresses in each of 2% of all allocated /24s are pinged repeatedly.

Other problems with the previous collection include measuring too few destinations per /24 to create significantly-sized test splits, failing to filter out destinations falling within private or restricted address ranges, and never re-trying failed probes. Our present run probes 1.5x more /24s and 2x IPs per-/24, filters out private and restricted ranges during selection, "canary-tests" all selected networks with a small number of probes from each VP to detect any rate-limiting early on, and retries unresponsive targets several times before giving up.

In addition to these fixes, we are collecting ICMP-traceroutes to a subset of the selected destinations from each /24 from all VPs. The purpose of this is twofold: to enable more complete case-studies, and to conduct an auxiliary analysis of the prevalence of the BCP-186 "forwarding-without-stamping" policy.

Vantage Point Augmentation. One avenue by which Reverse Traceroute may be made even more scalable is via using Tor's relay network to increase the available number of VPs. Currently, we have to use nodes from M-Lab, Planet Lab, RIPE Atlas, etc. as they are "trusted" sources on the internet and are allowed to spoof source IP addresses without being subjected to packet drops. Since Tor encrypts the entire IP packet repeatedly and does not reveal source IP address either to any of the hops on intermediate relays, we might be able to effectively make any personal computer a Vantage Point in the world; thereby increasing our set of available VPs and providing much needed variance for these VPs. It must be noted, however, that this method may be severely hampered by the tendency for spoof-filtering to exist near the edge [6].

5 Conclusions

In this work we have walked through the profiling and critique of several strong candidate algorithms for dynamically selecting Vantage Points in the online Reverse Traceroute system. The precursor to this work proposed *Ingress-Cover* which takes advantage of the fact that paths tend to converge as they move closer to destinations.

When our evaluation pinpointed several major drawbacks of this approach, careful examination of and iteration through multiple case-studies inspired an improved ingress-based approach: *Ingress-Scoring*. By eliminating aggregation, expanding ingress definitions, and adding a weighted distance-plus-frequency heuristic, our new approach outperforms every other algorithm candidate in all metrics. *Ingress-Scoring* enables the discovery of over 50% more reverse-hops

than the next-best method, while remaining RR-reachable to *at-least* as many destinations, and adding only a negligible additional number probes.

We issued real measurements, split them into test and training sets, and used them for emulating an iteration of ranking VPs by each method and testing their efficacy. The results of this show ingress-scoring to be a major improvement over its predecessor, and makes a strong case for a push to production. After closing our discussion of which Vantage Point selection strategy to use, we intend to re-demonstrate the system's effectiveness and upgraded scalability through several large applications. Two areas of immediate interest are (1) measuring the degree of path-asymmetry in the Internet, and (2) leveraging RR-reachable hosts as de-facto traceroute servers to augment the reach of high-level routing oracles like Sibyl[4].

References

1. Reverse traceroute web gui, `https://revtr.ccs.neu.edu`
2. Measurement lab (May 2019), `https://www.measurementlab.net/`
3. Calder, M., Fan, X., Hu, Z., Katz-Bassett, E., Heidemann, J., Govindan, R.: Mapping the expansion of google's serving infrastructure. In: Proceedings of the 2013 conference on Internet measurement conference. pp. 313–326. ACM (2013)
4. Cunha, Í., Marchetta, P., Calder, M., Chiu, Y.C., Machado, B.V., Pescapè, A., Giotsas, V., Madhyastha, H.V., Katz-Bassett, E.: Sibyl: a practical internet route oracle. In: 13th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 16). pp. 325–344 (2016)
5. F. Gont, R. Atkinson, C.P.: Recommendations on filtering of ipv4 packets containing ipv4 options. RFC 7216, IETF (February 2014), `https://tools.ietf.org/html/bcp186#section-4.5`
6. Goodchild, B.J., Chiu, Y.C., Hansen, R., Lua, H., Calder, M., Luckie, M., Lloyd, W., Choffnes, D., Katz-Bassett, E.: The record route option is an option! In: Proceedings of the 2017 Internet Measurement Conference. pp. 311–317. ACM (2017)
7. Govindan, R., Paxson, V.: Estimating router icmp generation delays. In: Passive & Active Measurement (PAM) (2002)
8. Katz-Bassett, E., Madhyastha, H.V., Adhikari, V.K., Scott, C., Sherry, J., Van Wesep, P., Anderson, T.E., Krishnamurthy, A.: Reverse traceroute. In: NSDI. vol. 10, pp. 219–234 (2010)
9. Krishnan, R., Madhyastha, H.V., Srinivasan, S., Jain, S., Krishnamurthy, A., Anderson, T., Gao, J.: Moving beyond end-to-end path information to optimize cdn performance. In: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement. pp. 190–201. ACM (2009)
10. Maynard-Koran, P.: Fixing the internet for real time applications: Part i, `https://engineering.riotgames.com/news/fixing-internet-real-time-applications-part-i`