DisCarte: A Disjunctive Internet Cartographer*

Rob Sherwood
University of Maryland
capveg@cs.umd.edu

Adam Bender
University of Maryland
bender@cs.umd.edu

Neil Spring
University of Maryland
nspring@cs.umd.edu

ABSTRACT

Internet topology discovery consists of inferring the inter-router connectivity ("links") and the mapping from IP addresses to routers ("alias resolution"). Current topology discovery techniques use TTL-limited "traceroute" probes to discover links and use direct router probing to resolve aliases. The often-ignored record route (RR) IP option provides a source of disparate topology data that could augment existing techniques, but it is difficult to properly align with traceroute-based topologies because router RR implementations are under-standardized. Correctly aligned RR and traceroute topologies have fewer false links, include anonymous and hidden routers, and discover aliases for routers that do not respond to direct probing. More accurate and feature-rich topologies benefit overlay construction and network diagnostics, modeling, and measurement.

We present DisCarte, a system for aligning and cross-validating RR and traceroute topology data using observed engineering practices. DisCarte uses disjunctive logic programming (DLP), a logical inference and constraint solving technique, to intelligently merge RR and traceroute data. We demonstrate that the resultant topology is more accurate and complete than previous techniques by validating its internal consistency and by comparing to publicly-available topologies. We classify irregularities in router implementations and introduce a divide-and-conquer technique used to scale DLP to Internet-sized systems.

Categories and Subject Descriptors

C.2.1 [**Communication Networks**]: Network Architecture and Design — Network Topology

General Terms

Measurement, Experimentation, Verification

Keywords

Network Topology Discovery, DisCarte, Disjunctive Logic Programming, Record Route, Alias Resolution

*This work was supported by grants ANI 0092806 and CNS-0435065 from the National Science Foundation.

SIGCOMM'08, August 17–22, 2008, Seattle, Washington, USA.

1. INTRODUCTION

Knowledge of the global topology of the Internet allows network operators and researchers to determine where losses, bottlenecks, failures, and other undesirable and anomalous events occur. Yet this topology remains largely unknown: individual operators may know their own networks, but neighboring networks are amorphous clouds. The lack of precise global topology information hinders network diagnostics [42, 24, 15, 17], inflates IP path lengths [10, 39, 36, 43], reduces the accuracy of Internet models [46, 25, 16], and encourages overlay networks to ignore the underlay [2, 27].

Because network operators rarely publish their topologies, and the IP protocols have little explicit support for exposing the Internet's underlying structure, researchers must infer the topology from measurement and observation. A router-level network topology consists of two types of features: links and aliases. A *link* connects two IP addresses on distinct routers, and an *alias* identifies two IP addresses on the same router. The goal is to discover a router-level map that is both accurate—all inferred features reflect the actual topology—and complete—features are inferred for as many pairs of IP addresses as possible.

The problem is that topology discovery techniques are error-prone. The current state-of-the-art [40, 21] uses TTL-limited probes, i.e., traceroute (TR), to infer links, and direct router probing [40, 12] to discover aliases. However, topologies inferred from these techniques are known to inflate the number of observed routers [44], record incorrect links [3], and bias router degree distributions [18]. These errors result from routers that do not respond to alias resolution techniques, anonymous routers [45], mid-measurement path instabilities [30], MPLS [35], and insufficient measurement vantage points. The Passenger tool [38] demonstrates that the record route (RR) IP option discovers aliases for unresponsive routers and exposes MPLS tunnels, anonymous routers, and mid-measurement path instabilities. However, RR's accuracy depends on correctly aligning RR and TR discovered IPs, itself an error-prone procedure. Passenger's preliminary work reports that almost 40% of their data could not be aligned and was unusable. Of the usable data, almost 11% of the inferred aliases were incorrect.

Unfortunately, accuracy and completeness can be at odds: for example, measuring more path data can help complete the map, but may also contribute inaccurate links. This is because topology errors accumulate—adding additional correct facts cannot "average away" a falsely asserted link. Similarly, alias inferences are transitive—a single false alias causes a cascading transitive closure of false aliases. We make the observation that if all topology data has error, then the vast data required for a complete map must have a great deal of accumulated error. Thus, in order to achieve both accuracy and completeness *at the same time*, a topology inference system must actively identify and remove error.

Our insight is that the overall error can be reduced by *cross-validating* both TR and RR inference techniques against observed network engineering practices. For example, a correctly implemented router would never forward packets directly back to itself, so any topology that asserts a link and an alias between the same pair of IP addresses must be inaccurate. Thus, by carefully merging three disparate sources of information, the resultant topology is both more accurate and more complete.

We present DisCarte, a novel topology data cross-validation system. DisCarte formulates topology inference and cross-validation as a constraint solving problem using disjunctive logic programming (DLP). DisCarte inputs traces from TR and RR, and, using observed network engineering practices as constraints, outputs a single merged topology. Compared to Rocketfuel-based [40] techniques, topologies produced with DisCarte find 11% more aliases from unresponsive routers, and expose additional topology features such as MPLS, router manufacturer, equal cost multi-pathing, and hidden and anonymous routers. Compared to Passenger [38], DisCarte correctly aligns 96% of RR and TR addresses, and reduces the false alias rate to approximately 3%. The effect of the improved topology is visually evident: we compare the topology of the popular Abilene network as inferred by Rocketfuel and DisCarte to the actual published topology (Figure 1).

In this paper, we describe the qualitative benefits of DisCarte inferred topologies (Section 2) and the difficulties in achieving accurate topologies (Section 3). We then discuss the individual elements of the DisCarte system (Section 4) and a novel divide-and-conquer scheme (Section 5) we implement to scale DLP to the 1.3 billion facts in our system. We detail our data collection process (Section 6), quantify the benefit of DisCarte inferred topologies (Section 7), and show DisCarte's effect on bias (Section 8). We then conclude how one might redesign record route (Section 10) to aid topology discovery and describe our future work (Section 11).

2. CROSS-VALIDATING WITH DISCARTE

In this section, we describe the benefits of correctly merged traceroute-inferred and RR-inferred topologies. Traceroute (TR) uses TTL-limited probes to generate ICMP time-exceeded responses from each router on a path. The source IP address of each time-exceeded message exposes an IP address for the corresponding router. The record route (RR) IP option is an array in the IP header into which each router on the path inserts an IP address. The array can store at most nine addresses, bound by the size limit of the IP header. Because of how they are implemented (Section **??**), TR and RR discover *distinct* IP addresses for a given router. TR discovers the IP address for the *incoming* interface whereas RR can discover the *outgoing* or *internal* routing interface depending on implementation. We say that a TR-trace and RR-trace have been correctly *address aligned* if each TR-discovered address has been correctly mapped to the RR-discovered address of the same router.

TR and RR can be combined into a single TTL-limited probe with the RR option set. Because an ICMP unreachable error message includes the entire IP header of the failed message, we can recover the RR array from the responses to TTL-limited probes: RR packets need not reach the destination of the probe. Thus, RR does not require the packet destination to return the RR probe, i.e., "ping -R" is not the only means of collecting RR data.

2.1 Benefits of Cross-Validation

Cross-validating TR and RR information against observed network engineering practices results in higher quality address alignment. Correct address alignment discovers aliases for routers that do not respond to direct probing, hidden and anonymous routers, and multi-path load balancing.

Alias resolution does not require direct probing.

In our survey, 193,192 of 602,136 (32.1%) IP addresses do not respond to probes addressed directly to them, preventing both IP-identifier-based matching ("ally" [40]) and source-address matching [29, 12] alias resolution techniques. Six years ago, approximately 10% were unresponsive [40], suggesting that techniques for alias resolution without direct probing will be increasingly important. We further characterize the aliases RR allows us to discover in Section 7.

RR exposes hidden and anonymous routers.

We call routers *hidden* if they do not decrement TTL and do not appear inside a traceroute; some implementations of MPLS [35] cause hidden routers. *Anonymous* routers [45] are routers that decrement TTL but do not send the corresponding ICMP time-exceeded messages: they appear as a '*' in traceroute. The absence of information from these routers is a significant source of error [45]. Out of 100,256 routers observed in our study, RR discovered IP addresses for 2,440 (2.4%) distinct anonymous routers that would have been missed by TR-only techniques. Additionally, we discover 329 (0.3%) distinct hidden routers.

RR discovers multi-path load balancing.

Internet Service Providers (ISPs) use multiple routes across equal-cost paths to load balance traffic. To prevent out-of-order packet arrival, load balancing routers attempt to map packets in the same flow to the same path. However, due to implementation decisions [6] in some routers, packets with IP options, including RR, break this flow-to-path mapping and traverse multiple equal-cost paths. Thus, probes with RR detect load balancing routers and enumerate additional paths more correctly than TR alone.

RR exposes mid-measurement path instability.

Mid-measurement path instabilities cause TR to infer incorrect links: TR assumes that sequential probes traverse the same paths. Recent techniques (Paris traceroute [3] and TCP Sidecar [37, 38]) mitigate this concern by preventing a specific class of instabilities: five-tuple load balancing multi-path. Because RR has per-packet path information, we can detect all forms of mid-measurement path changes in the first nine hops. Thus, links discovered via RR exist with higher confidence than links discovered by TR alone.

A trace between Zhengzhou University, China to SUNY Stony Brook, USA (Figure 2) is an example of the differences between topology discovery with and without DisCarte. Each box represents a router, and each rectangle within a router represents an interface. Lines between interfaces indicate links. We resolve DNS names of IP addresses when available, and show only the first four hops of the trace then a dotted line to the destination. The trace without RR discovers at most one interface on each router, and fails to discover any interfaces on router 3 (because it is anonymous). Adding RR to the probes and performing address alignment (Section 3) discovers many interfaces on each router and exposes many connections between routers, presumably for load balancing. Router labels (S1, R2, etc.) are annotated with their inferred RR implementation type (Section 3.1).

2.2 Cross Validation Limitations: RR

Many of the benefits of cross-validation rely on the RR option which has two limitations: RR includes only nine hops of data and packets with RR may be dropped or filtered. We describe each in turn.

Because the IP header can hold at most 60 bytes, RR can record only nine IP addresses. We believe the nine-hop limit is why RR has been passed over for topology discovery. Yet, there is reason

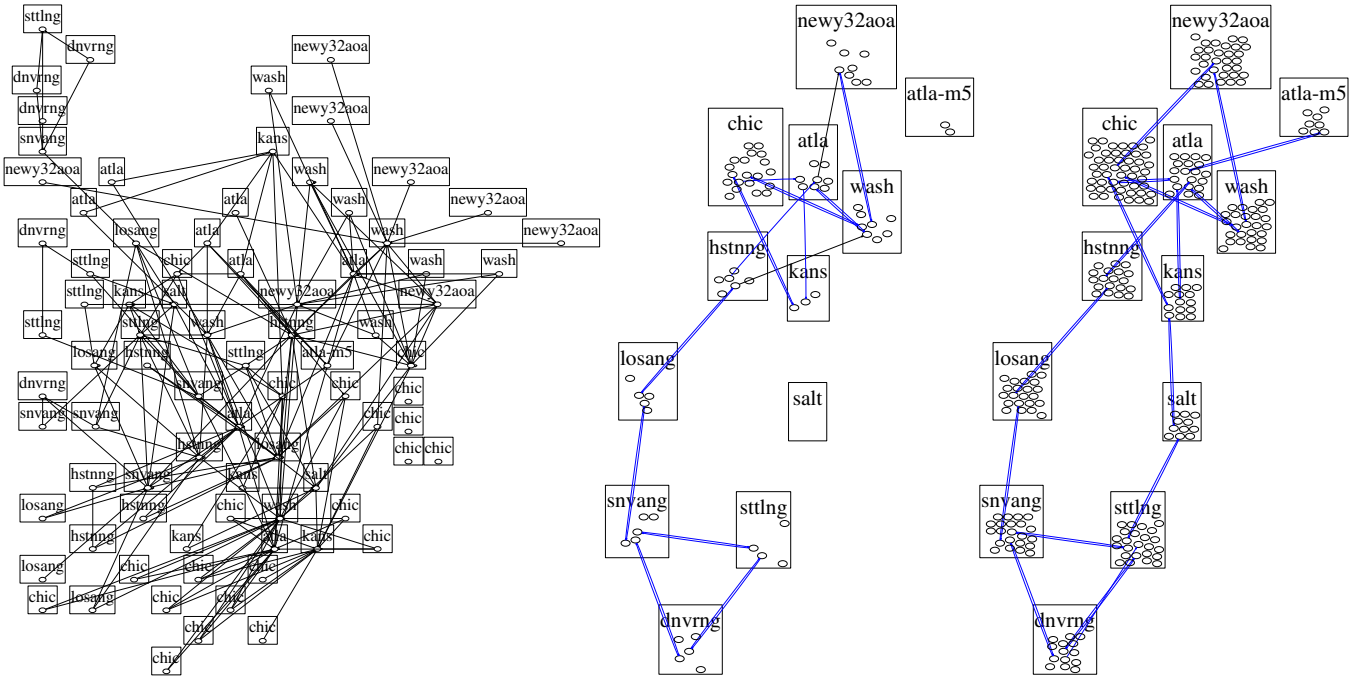

Figure 1: Abilene topology: inferred by Rocketfuel (left, routers unresponsive to direct alias resolution), DisCarte (middle), and actual topology (right). Rectangles are routers with interior ovals representing interfaces.

to revisit this concern. PlanetLab makes available a geographically diverse set of vantage points; these may be within nine hops of much of the network. Further, our experiments use TR probes with and without RR set, so any information gained from RR strictly increases our understanding of the topology.

Second, routers might choose to drop or filter packets with IP options. Of the 602,136 IP addresses of routers we observed within nine hops of our vantage points (that could have dropped RR), only 8,441 (1%) dropped packets with record route. We mitigate this limitation by running all traces with and without RR set.

3. ADDRESS ALIGNMENT

In order to achieve the benefits of cross-validation (Section 2), addresses discovered by TR and RR must be correctly *aligned*. Address alignment is the process of matching the IP addresses discovered by RR to the corresponding addresses discovered by TR. Accurate address alignment requires classifying the RR implementation type of each router in a trace and correctly handling tricky topology features.

3.1 Under-Standardized RR Implementations

The record route IP option [33] tells routers to record their IP address into a buffer in a packet's IP header. The interface that is recorded is the first source of implementation variation. Although RFC 791 states that a router should record "its own Internet address as known in the environment into which this data-gram is being forwarded," we have observed that routers record the address corresponding to the incoming, outgoing, or internal interface depending on implementation. The second implementation variation we have observed is whether the address is recorded for an expiring packet, that is, when a packet arrives with TTL=1.

We observe six different RR implementations. We describe each implementation, sorted in order of popularity, along with our best estimate of its manufacturer. An implementation's popularity is a function of the total number of routers we were able to classify.

Departing: 61.9% This implementation updates the RR array as packets leave the router, and thus does not update the RR array for expiring packets. That is, when a TTL=1 packet arrives at a router, the router does not add an address to the RR array. When a packet with TTL>1 passes through the router, the *outgoing* interface address is recorded. We associate this behavior with Cisco routers due to its popularity and private communications with Cisco engineers.

MPLS: 13.3% This implementation behaves like a Departing router (above), except for interfaces with MPLS [35] enabled. A packet that exits an MPLS-enabled interface does not modify the RR array (similar to NotImpl, below). We know that these interfaces use MPLS because they also implement the ICMP unreachable MPLS trailers protocol [4] that returns MPLS tunnel identifiers.

NotImpl: 9.1% Some routers disable or do not implement RR. These routers pass RR probes through without modification. We believe that RR may have been previously overlooked as a measurement technique due to inflated expectation of the number of NotImpl routers.

Arriving: 7.1% In this implementation, the RR array is updated when the packet arrives so TTL expiring packets are updated. Some routers record the outgoing interface, while others record the internal loopback interface. Internal loopback addresses can be distinguished from outgoing addresses by hand, for example, if the reverse DNS look-up of the address contains the string "lo-". We believe this RR-type corresponds to Juniper due to its appearance in the Abilene network which uses Juniper routers [1].

Lazy: 5.8% These routers do not decrement TTL for packets with the RR option set, and instead allow the packet to continue

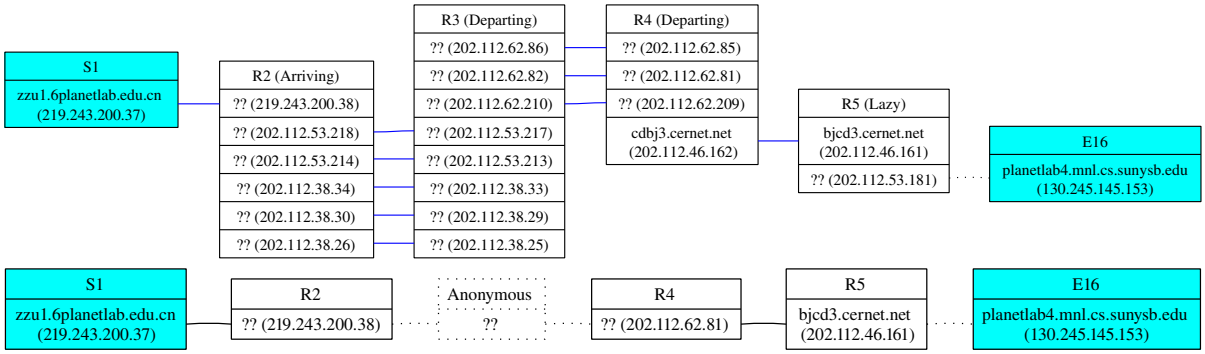

Figure 2: Partial Trace from Zhengzhou University, China to SUNY Stony Brook, USA; inferred by DisCarte (top) and Rocketfuel techniques (bottom). DisCarte finds many load-balanced paths through an anonymous router (R3) and helps determine the implementation class of each device along the path.

to the next hop. This caused significant confusion in our initial experiments using interleaved packets with and without RR set. Publicly available router configurations at National LambdaRail (NLR) suggest that Cisco's Carrier line of routers are Lazy. Of all RR implementations we have observed, this is the only one that would seem to violate RFC 791.

Mixed: 2.7% Some routers have mixed behavior for arriving and departing packets. If the packet arrives and expires, the router updates the RR array with the *incoming* interface address. Else, if the packet does not expire, the router updates the RR array with the *outgoing* interface address. We believe that Linux-based IP stacks implement this behavior.

With the exception of the Lazy RR implementation, we believe that these implementation variations correctly implement the RR specification as described in RFC 791. The variations in implementation arise because RR is underspecified, and we recommend additions to the specification (Section 10). Also, note that the "Flaky" RR implementation, first identified by Sherwood and Spring [38], does not appear to exist. We believe that Flaky is a combination of the Lazy implementation type above and equal-cost path routing of different hop counts.

3.2 Topology Traps

We identify six topology features that complicate accurate topology discovery. In this section, we catalog these features to show the complexity inherent in topology discovery and motivate the need for an automated inference tool.

Hidden routers do not decrement TTL and thus are not detected by TTL-limited topology discovery. Hidden routers are caused by certain configurations of multi-protocol label switching [35] (MPLS) and result in missing nodes and incorrect link inferences. As with anonymous routers, the RR IP options can be used to detect hidden routers if supported. Also, the use of MPLS can be detected by an optional MPLS tag attached as a footer in TTL-exceeded messages [4]. We discovered 329 hidden routers in our experiments.

Non-standard firewall policies introduce varied sources of error. In one case, a firewall in China forges TTL-exceeded messages from the destination [38] for packets with the RR option set. Also, we have observed firewalls that send ICMP source quench, ICMP parameter problem, and ICMP administratively prohibited messages. Each of these behaviors must be identified and removed from the data before processing.

Enabling IP options breaks load-balancing, spreading a single flow across multiple equal-cost paths. Five-tuple load-balancing uses the source and destination IP and port fields along with the IP protocol to identify a flow and maps all packets in the same flow to the same path [3]. However, adding IP options to packets with the same five-tuple signature breaks this scheme. We hypothesize that some router implementations fail to account for IP options when calculating the packet offset to the TCP/UDP source and destination port fields when computing the 5-tuple. In other implementations, packets with IP options are routed on arbitrary equal-cost paths. Both behaviors add to the complexity of address alignment.

Different-length equal-cost paths can create false links and aliases. Equal-cost paths may have different hop-count lengths, which results in multiple sets of probes, offset in TTL, between the same source and destination. Comparing probes from different paths may cause false topology assertions, e.g., not all routers at TTL=3 have a link to routers at TTL=4. We use RR to partition probes by the path they traversed, and only compare probes that take the same path. By partitioning probes by the paths that they traverse, we remove one source of self-loops common to traceroute-inferred topologies [3]. Traces from Cornell University to PlanetLab nodes in Amsterdam have this behavior (Figure 3).

RR fills. The address alignment algorithm monitors hop-by-hop increases in the size of the RR array to classify each router's RR type (Section 4). Because a given hop may add more than one entry into the RR array when the RR array fills up—reaches nine entries—the information about the true number of RR entries for this hop is lost. For example, a packet with eight RR entries that transitions from a Departing RR-type router to a Arriving RR-type router, would normally receive two new RR entries. However, since there is only space for one more IP address, the second entry is lost. The address alignment algorithm has to consider more possibilities when the RR array fills. DisCarte's DLP code base doubles in size to handle this seemingly simple case.

Persistent Routing Loops can prevent naïve trace collection from terminating. Our data collection scripts had to be rewritten to detect loops. We revisited the looping paths three weeks later and found that approximately half still persisted. In Section 6.3, we further characterize the routing loops we discovered.

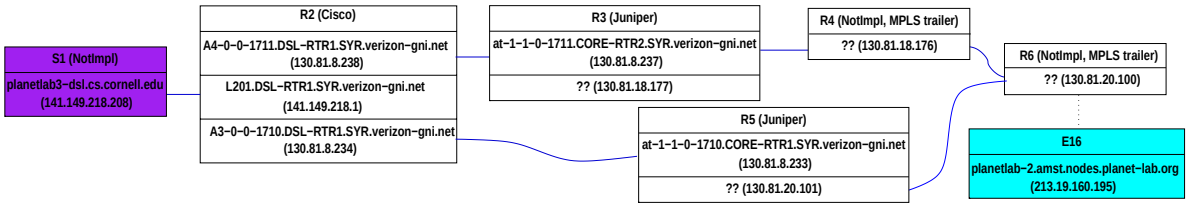

Figure 3: Partial trace from Cornell to Amsterdam where probes that take different-length paths: bottom path is one hop shorter.

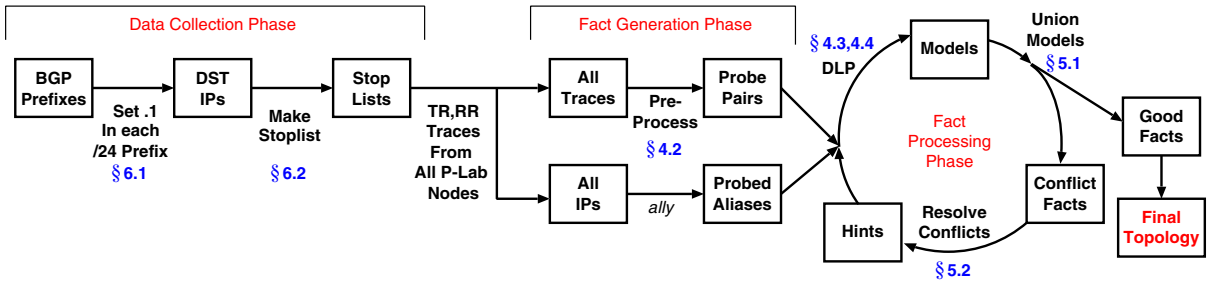

Figure 5: Overview of the DisCarte Topology Inference System.

3.3 Ambiguity in classification

The variety of RR implementations make router classification ambiguous. Because different topologies and router implementations can generate the same trace (Figure 4), a router may be misclassified, leading to mismatched addresses and aliases. Thus, in the same trace, IP X might be an alias for IP A or B depending on the RR implementation. Further, the third probe discovers two new RR addresses (Y, Z) and it is ambiguous whether IP address Y belongs to a hidden router. We depict two of 15 possible interpretations of the trace.

A single mismatched pair of addresses causes cascading errors as each subsequent RR address in a trace is misaligned. However, using observed network engineering practices it is possible to correctly match RR and traceroute discovered addresses (Section 4.4). For example, network engineers tend to allocate IP addresses on either end of a link out of a /30 or /31 network [14, 21], so the topology that best matches this pattern is most likely correct.

Probe ttl	ICMP source IP	RR Array
1	A	-
2	B	X
3	C	X, Y, Z

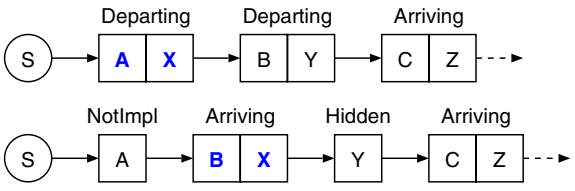

Figure 4: Varied RR implementations create ambiguous alignments between IP addresses discovered by TR (A, B, C) and those discovered by RR (X, Y, Z). We show two of 15 possible topologies inferred from a hypothetical trace from source S: rectangles represent routers and letters are IP interfaces.

4. DISCARTE

Large-scale cross-validation and address alignment is difficult and error-prone, not only because of the need to infer the different RR implementations of routers, but also due to complex network topology features (Section 3).

Our system, DisCarte, uses disjunctive logic programming (DLP) [5, 34, 19], a constraint solving technique that, to the best of our knowledge, has not been used for topology discovery. DLP has the ability to describe a low-level set of inter-dependent interactions while simultaneously shaping the solution to match high-level constraints. For example, we instruct DLP to find the set of RR implementations such that the link and alias assignments do not cause routers to have self-loops. The DisCarte process (Figure 5) consists of fact generation (Section 4.2) and fact processing (Section 4.3 – 5) phases. We describe our data collection phase in Section 6. In this section, we provide a brief description of the DLP technique, describe how we transform raw topology data into DLP facts, and present the DisCarte address alignment algorithm and its corresponding cost function.

4.1 DLP Introduction

DLP is a formalism representing indefinite information. Superficially similar to Prolog, language statements consist of facts, inference rules, and weak and strong constraints. Inference rules are *disjunctive*—they are of the form:

$$\text{fact}_1 \textbf{ or } \text{fact}_2 \textbf{ or } \ldots \textbf{ or } \text{fact}_n \Leftarrow \text{fact}_0 \qquad (1)$$

indicating that fact_0 implies exactly one fact in the set of facts $\text{fact}_1 \ldots \text{fact}_n$. Because each inference rule can potentially imply many different facts, a disjunctive logic program has many possible solutions, or *models*. Potential models are then pruned by strong and weak *constraints*. Any model that violates a strong constraint is removed from the solution set, and the remaining models are assigned a numeric cost based on the weak constraints they violate. The output from a DLP is the lowest cost model of inferred facts generated from input facts and inference rules.

The specific DLP implementation we use is DLV [19]. The language restricts how constraints are specified, to preserve the *mono-*

tonicity property of the cost function: that adding new facts can only increase that cost of a model. DLV uses this property to prune high cost sub-trees from the solution space. DLP can efficiently represent complex problems, for example, the formulation for the graph 3-color-ability problem [11] is two lines long [8].

4.2 Data Pre-processing

Raw trace data must be converted into facts for DLP. These facts consist of both straightforward parsing of the data, deriving facts more easily computed without DLP, and *probe pairs*. Here, we describe the facts computed in the pre-processing step.

Some network topology features can be identified statically without DLP. Routers with the Mixed RR implementation have a simple signature: the response to a TTL-limited probe comes from router X, and the last entry in the RR array is also X. Similarly, we declare a router X to be Lazy if all non-RR probes with TTL=t return ICMP time-exceeded responses from X, all RR probes with TTL=t return responses from a different router Y, and all non-RR probes with TTL=$t+1$ return from router Y. Responses to RR probes from non-standard firewalls have the source address set to the probe's destination, instead of the router's interface. Once these network features have been detected, we correct for them as we identify probe pairs.

Two TTL-limited probes form a "probe pair" if one probe expires at router X, and the other probe goes through X and expires at the next TTL. Each probe pair fact is of the following form: "probePair(p_1,p_2,$delta$)", where p_1 and p_2 are unique probe identifiers, and $delta$ is the difference between the size of the two RR arrays. By convention, p_2 is the probe that went one TTL farther. Identifying probe pairs in non-RR (traceroute-only) data is trivial but error prone: mid-measurement path instabilities (Section 2) can cause sequential probes to take different paths. When adding RR to probes, probe pair identification becomes more accurate—RR probes record the traversed path—but more complicated. Lazy RR implementations and multi-path routing with different length paths complicate probe pair identification. For example, after passing through a Lazy router, TTL-limited probes with RR set go one hop farther then intended. Before we can try to identify probe pairs in the presence of a Lazy router, all probes with RR that pass through that router must be re-normalized as if they were sent from the subsequent TTL. Also, if there is evidence of multi-path routing with different length paths, we must be careful to only compare probes that took the same length path. Last, if a trace has both Lazy routers and different length paths, we can only identify Lazy routers on the path taken by the non-RR probes, so information on the other path must be discarded.

4.3 Address Alignment with DLP

Though an exotic choice, DLP lends itself well to the address alignment problem. For each trace, the pre-processor will output a set of potentially over-lapping probe pairs: probePair($X, Y, delta_1$) and probePair($Y, Z, delta_2$). The job of the DLP is to infer the most likely RR implementation type assignments that are globally consistent: router Y must have the same RR type in all of its probe pairs. Then, based on the type assignments, DLP outputs link and alias facts that form a topology.

To constrain the assignment of implementation types to those consistent with the probe pair facts, we express DLP inference rules that describe each possible transition from router to router for each $delta$ in a probe pair. The $delta$ is the number of additional RR entries in the second probe of the pair, and may be any number from 0 to 9, though we have not observed a delta greater than 4. An RR entry will be added when (a) leaving a Departing router, (b) arriving at an Arriving or Mixed router, or (c) traversing a Hidden router. Traversing a NotImpl router (or entering an MPLS tunnel, which has the same effect) does not add to the delta. The inference rule is the list of possible RR-type transitions that would result in a probe pair with the same $delta$. For example:

$$\begin{aligned} transition(X, Y, Departing, Departing) \quad & \textbf{or} \\ transition(X, Y, Arriving, Arriving) \quad & \textbf{or} \\ transition(X, Y, Departing, NotImpl) \quad & \textbf{or} \\ transition(X, Y, NotImpl, Arriving) \quad & \textbf{or} \\ transition(X, Y, NotImpl, Hidden, Departing) \quad & \textbf{or} \\ transition(X, Y, NotImpl, Hidden, NotImpl) \quad & \textbf{or} \\ \Leftarrow probePair(X, Y, delta), & \\ delta = 1. & \end{aligned} \tag{2}$$

indicates that if we find a probe pair with $delta = 1$, then the transition between the router corresponding to the first probe (X) and the router corresponding to the second probe (Y) is a transition from a Departing-type RR router to another Departing-type RR router *or* from a Arriving-type RR type router to another Arriving-type RR router, etc. We must include more atypical transitions, such as from a router that does not implement RR (NotImpl), through a router that does not show up in traceroute but implements RR (Hidden), to another router that does not implement RR (NotImpl). We wrote DLP inference rules for delta=$0 \ldots 4$, and show the possible transitions from 0 through 2 in Table 1. We also implement a duplicate set of all rules where the RR array is full (Section 3.2). Thus for a probe with full RR array and $delta = X$, all possible transitions for $delta \geq X$ must be considered.

Multiple possible transitions per probe pair and independent computation of probe pairs imply that there are potentially exponentially many models relative to the number of probe pairs. We discuss the cost function for intelligently pruning this set to produce the best model (Section 4.4) and our divide-and-conquer technique for scaling this algorithm (Section 5).

4.4 Engineering Practices and Cost Function

Recall from Section 4.1 that DLP supports strong and weak constraints: models that violate strong constraints are removed and the rest are ordered by degree of weak constraints violated. DLP outputs the lowest cost model.

The only strong constraint in the DisCarte system is that a router's RR implementation must be consistent across all its interfaces. In other words, it is never the case that the same router uses the Departing RR behavior for one interface and Arriving RR behavior for another interface. A potential issue with this rule is the MPLS RR type, where individual interfaces might appear to be of RR type Departing or NotImpl. The strong constraints are carefully written to handle this exception.

Weak constraints are chosen based on observed patterns which we believe correspond to network engineering practices. Each practice should hold as a general rule of thumb, but may be violated in an individual solution. Thus the model that violates the fewest practices is likely to be the closest approximation of reality. Here we list weak constraints in order of importance.

1. There should be no self-loops: a correctly-implemented router would never route packets directly back to itself. Avoiding this condition prevents situations where two distinct routers are merged by a single bad alias, and conversely when a link is incorrectly added between interfaces on the same router [3].
2. Many IP addresses on either end of a link are adjacent in IP space: they are "off-by-one." We expect that network ar-

delta=0	delta=1	delta=2
NotImpl → NotImpl	NotImpl → Hidden → NotImpl	NotImpl → Hidden → Hidden → NotImpl
NotImpl → Departing	NotImpl → Hidden → Departing	NotImpl → Hidden → Hidden → Departing
Arriving or Mixed → NotImpl	Arriving or Mixed → Hidden → NotImpl	Arriving or Mixed → Hidden → Hidden → NotImpl
Arriving or Mixed → Departing	Arriving or Mixed → Hidden → Departing	Arriving or Mixed → Hidden → Hidden → Departing
	NotImpl → Arriving or Mixed	NotImpl → Hidden → Arriving or Mixed
	Departing → NotImpl	Departing → Hidden → NotImpl
	Departing → Departing	Departing → Hidden → Departing
	Arriving or Mixed → Arriving or Mixed	Arriving or Mixed → Hidden → Arriving or Mixed
		Departing → Arriving or Mixed

Table 1: Possible router RR implementation transitions arranged by RR delta; deltas 3 and 4 are not shown. Arriving and Mixed are written together to save space.

chitects try to conserve address space by using the smallest network blocks available, either /30 or /31. The implication is that models where the IP addresses of links are off-by-one should be preferred over those without. Gunes et al. use this technique to infer aliases directly [14]. Figures 2 and 3 show this behavior.

3. Aliases inferred by direct probing (ally [40]) are often correct. The validity of direct probing techniques [41, 12] has been independently demonstrated, so that information should be used when available. However, due to temporal changes in topology or potential for inaccuracies in the technique, information from direct probing remains a weak constraint.

4. Hidden routers are rare, so of two equally-likely models, the solution with the fewest hidden routers should be preferred. We derive this rule from observation of out-of-band data, such as DNS naming conventions and /30 and /31 IP addressing in links.

5. Routers supporting RR are more common than those that do not (NotImpl). We verify this empirically by observing that with each new TTL, subsequent probes in a trace typically record new RR entries.

The cost for a model is assigned based on the number of practices violated, weighted by the importance of the practice. We experimented with different weight assignments, but as long as the relative importance of practices remained as above, the weight assignment did not affect the final solution. Also, it is possible for DLP to output multiple equal-cost models, if there is insufficient information to make an alignment, or no model at all, if there is an error in the data or flaw in our model. We next address both points further.

5. SCALING AND CONFLICTS

DLP alone does not scale to Internet-sized topologies, as the number of possible RR implementation assignments grows exponentially with the number of probe pairs. Our top-level approach is to process the data in pieces large enough to provide the correct solutions, yet small enough that they are solved quickly—divide and conquer. Merging processed pieces back together can expose *conflicts*: that the same pair of IP addresses are believed to be both aliased and linked. In this section, we describe a data partitioning method that reduces conflicts and engineer a technique to resolve conflicts once they occur.

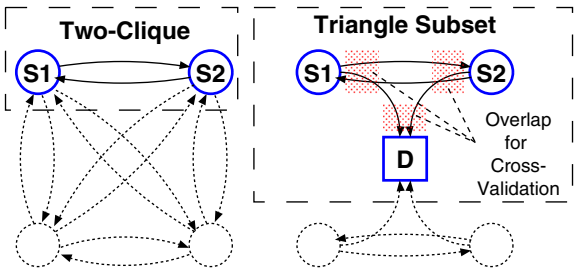

Figure 6: We first align addresses in two-cliques (left) between all sources and then subset triangles (right) to all destinations increasing overlap and decreasing errors.

5.1 Divide and Conquer

Dividing the data is easy; dividing the data while preserving enough information for DLP to produce meaningful results is difficult. Our first approaches at partitioning the data produced a scalable execution—one trace per run, or many traces from the same source—but they resulted in many incorrect inferences. Because each run interpreted only the data from probes leaving the source, the DLP solver missed potentially conflicting data from measuring the return path.

To provide a core of correct, reliable address alignments and router implementation inferences, we start by computing all two-cliques—the trace from site X to Y with the trace from Y to X—as shown in Figure 6, left.

Atop this core, we process triangle-like subsets of all traces between pairs of sources and a destination (Figure 6, right). The insight is that the path between the source pair has already been computed and found to be free of conflicts, so it is reliable. By using this approach, we reduce the number of unresolved conflicts—those conflicting inferences that remained after all processing—from 1,547 to 28 in the PlanetLab data set.

We hoped to process all possible triangle subsets for maximum overlap and thus maximum cross-validation, but with 379 sources and 376,408 destinations, this task is intractable. Instead, we processed the 71 million non-overlapping triangle subsets on a 341 processor (heterogeneous) Condor [20] cluster. Triangle subsets typically take a second to process, though the execution time is highly variable. The Condor scheduler estimates that we have used 96,225 hours or approximately 11 CPU years on this project (including time spent debugging).

5.2 Unions and Conflicts

We extract the facts in the models produced by the divide and

conquer phase and search for contradictions. A contradiction appears when two addresses that are thought to be aliases are seen to be linked in a subset of facts. (Two IP addresses can be assigned to the same router if they are aliases of aliases, so the alias inference can result from several sets of facts; a link cannot be synthesized from different traces–see Section 4).

To resolve conflicts, we pick an arbitrary model from each faction (those indicating link and those indicating alias) and run both input subsets together through DLP. If the result contains exactly one model, then the conflict is resolved, and we record whether the IP addresses are linked or aliased as a *hint*. Once the hint is recorded, all affected models are recomputed via DLP.

The conflict resolver can fail to resolve a conflict if the DLP outputs multiple models with both link and alias facts asserted, or no model at all. Having multiple models indicates that we have insufficient information to resolve this conflict, whereas producing no models indicates an error in the input or a potentially new RR behavior. In any case, if the conflict resolver cannot resolve the conflict, then all facts associated with the two IP addresses are removed from the model. In our experiments, 12,731 of 9,793,309 (0.13%) of subsets produce no valid model, and 22,095 of 1,021,027 (3.7%) of facts remain unresolved. It is the subject of our future work to characterize the unresolvable traces and improve the conflict resolution process.

6. DATA COLLECTION

We collect two sets of topology data to validate DisCarte: one between PlanetLab nodes and the other from PlanetLab nodes to all advertised BGP prefixes. For both, we perform TTL-limited traceroute-like probes with and without the RR option set. For the BGP prefixes data set, we also use the "stoplist" technique to avoid probing destinations in a way that might appear abusive. We conclude by reporting on the distribution of stable routing loops that we discover in our experiments.

6.1 Data Sets

The PlanetLab [32] data set is an all-pairs trace, from all PlanetLab nodes to all other PlanetLab nodes. This repeats the Passenger [38] study. Because some PlanetLab nodes were unavailable, we were able to collect data from only 387 nodes.

In the BGP data set, we probe 376,408 destinations. To generate the destinations, we divide each advertised BGP prefix [26] into a /24 sub-prefix, choose a representative address from each by setting the last bit, and then remove unresponsive IP addresses. This IP generation strategy is similar to iPlane [21], except that we disaggregate larger prefixes down to /24-sized sub-nets.

We probe using traceroute's increasing TTL, alternating probes with and without the RR option set, three times with each. We stop probing a destination after probes for six sequential TTLs have been dropped. Due to firewalls that drop probes and source nodes rebooting, we do not have data for all sources to all destinations, but we do collect approximately 1.3 billion probe responses.

6.2 Stoplist Probing

We believe that RR probes are more likely to generate abuse reports then other topology discovery techniques. The RR option is rare and intrusion detection systems target anomalous events. However, we note that network mapping need not probe destination hosts often: careful measurement coordination can avoid reports of abuse. Our insight is that we can avoid probing destination networks from every source by noticing when the path from a new source merges with an already-observed path.

The goal of our "stoplist" technique is to give each destination a *red zone*: a region close to the destination that will not be probed from machines outside our control. A stoplist is a per-destination list of the last k IP addresses on a trace to the destination. We generate the stoplist from a single host under our administrative control, so potential abuse reports can be handled locally without involving PlanetLab support. To generate the stoplist, we run a reverse traceroute to each destination and record the last $k = 3$ hops. A reverse traceroute works by guessing the TTL distance to the end host, sending a probe, and then searching with larger TTL if the destination was not reached, or with smaller TTL if it was. Once the TTL distance to the end host is known, the last three hops are determined by decrementing the TTL.

The stoplist is then distributed among sources. As each node traces towards a destination, it stops when an IP on the stoplist is discovered. Using this technique, we received no abuse reports either in generating or using the stoplist.

6.3 Routing Loops

Routing loops are a symptom of network misconfiguration and can frustrate topology inference. DisCarte found a surprisingly high number of routing loops: 8,550 source-destination pairs contained a persistent routing loop which prevented packets from reaching the destination. These pairs were re-tested three weeks later. We were not able to retest 2,071 (24.1%) of these pairs, because the configuration of the source nodes had changed. At the later date, 4,501 (52.6%) of the loops were resolved, while 1,976 (23.1%) remained.

Of the routing loops that persisted through both tests, 689 unique routers appear 4,544 times in some part of a loop. China Railway Internet (CRNET, AS 9394) has more of these routers (61) than any other AS. Korea Telecom follows with 47 routers, and Level 3 with 35. When weighted by the number of traces that contain these loops, almost 10% of the routers again belong to CRNET, almost twice as many as the next-most frequent location, Frontier Communications of America (AS 5650).

7. VALIDATION

In this section, we validate the output of DisCarte in terms of accuracy and completeness. We first compare the aliases produced by DisCarte to those produced by Rocketfuel's ally tool [40]. Then, we compare the routers, links, and degree distribution of topologies inferred by DisCarte and the Rocketfuel and Passenger techniques against four published topologies.

Of course, any active IP-probing methodology will suffer from inherent shortcomings: that backup paths and link-layer redundancy are not visible, and that multiple-access network links are not differentiated from point-to-point links. DisCarte does not address these problems, so we do not consider them further.

7.1 RR Aliases

We use the IP-identifier [21, 40] and source-address matching [29, 12] alias resolution techniques to verify the aliases inferred by DisCarte. DisCarte over the BGP-prefixes data set found 374,337 aliases, 42,284 (11.2%) of which were not found by direct probe-based techniques in Rocketfuel's ally.

We then re-applied ally to confirm the aliases asserted by DisCarte: 88.3% were confirmed to be correct, 3.8% were claimed to not be aliases by the IP-identifier technique, and the remaining 7.8% were from unresponsive routers and could not be confirmed. 91.2% of the aliases found by DisCarte involved IP addresses discovered only by adding the RR option.

7.2 Comparison to Published Topologies

Research networks including Abilene[1], Géant[2], National LambdaRail (NLR)[3], and Canarie (CANET)[4] publish the configuration files of their routers, which makes determining a "correct" topology possible. We compare DisCarte's inferred map to these true topologies as well as the topologies produced by the Rocketfuel [40] and Passenger [38] techniques. To build the correct topology, we parse the "show interfaces" information available for each router from each network's web site. We use publicly available software to generate Rocketfuel[5] and Passenger[6] topologies. In each network, we consider the number and accuracy of discovered routers, the degree distribution, and completeness of the discovered links.

For each network, we classify each inferred router into one of four accuracy categories: good, merged, split, and missed.

Good: There is a one-to-one mapping between this inferred router and a router in the correct topology. All of this router's discovered interfaces are correctly aliased. In a perfectly inferred topology, all routers would be good.

Merged: There is a one-to-many mapping between this inferred router and routers in the correct topology. In this case, multiple real-world routers are incorrectly inferred as a single router. Merged routers result from inaccurate alias resolution, artificially deflate the router count, and inflate the node degree distribution.

Split: There is a many-to-one mapping between routers in the inferred topology and a single router in the correct topology. In this case, a single router from the correct topology appears split into multiple routers in the inferred topology. Split routers result from incomplete alias information, inflate the router count, and deflate the node degree distribution.

Missed: This router was not found: none of the router's interfaces were discovered by the inferred topology. Missing routers result from insufficient vantage points or from data discarded due to unresolved conflicts (Section 5.2). Missing routers deflate the router count and bias the node degree distribution towards observed routers.

Classifying the number of inferred routers by accuracy (Figure 7) illustrates three interesting characteristics. First, although all three inference schemes tend to have substantial numbers of "split" routers, Rocketfuel has so many split routers that it incorrectly over-estimates the router count by as much as seven times the true value. This is a result of routers that are unresponsive to direct alias probing, so no aliases are found (recall Figure 1). So, although aliases from DisCarte result in a more accurate topology, more complete alias resolution techniques are still required. Second, for each topology, DisCarte has more "good" nodes than other techniques, except for Passenger in the Géant network. In this exception, Passenger finds two more "good" nodes than DisCarte, at the cost of four incorrectly merged nodes. We demonstrate below that the presence of merged routers alters the topology's degree distribution. Third, DisCarte-inferred topologies have no merged routers and fewer split routers than Rocketfuel.

Next, we consider the degree distribution of inferred topologies. Degree distribution affects the accuracy of Internet-modeling [22] and path diversity studies [44], and has been studied in its own right [9]. We plot the degree distribution of the topologies inferred by Rocketfuel, Passenger, and DisCarte along with the actual degree distribution for each published topology (Figure 8). In all networks, the DisCarte inferred topology most closely tracks the actual degree distribution relative to the other two techniques. Also, the effect of merged routers on the degree distribution is apparent: Passenger deviates significantly from reality in the Géant data set due to the four merged routers it infers.

[1]http://vn.grnoc.iu.edu/xml/abilene/show_interfaces.xml
[2]http://stats.geant2.net/lg/process.jsp
[3]http://routerproxy.grnoc.iu.edu
[4]http://dooka.canet4.net
[5]http://www.cs.washington.edu/research/networking/rocketfuel/
[6]http://www.cs.umd.edu/projects/sidecar

	Abilene	CANET4	Géant	NLR
Links: Found	21	11	45	21
Total	33	16	62	22
(%)	63%	69%	72%	95%
False Links	0	0	0	0

Table 2: Completeness of DisCarte-inferred links.

Of four published topologies, our inferred topology has *no false links* (Table 2), and discovers at least 63% of existing links. We believe the only way to improve the completeness of the link coverage is to increase the number of measurement vantage points and their network diversity.

Comparison to research networks at first does not appear inherently challenging: their openness, homogeneity, and proximity to most PlanetLab vantage points make them relatively easy validation cases. However, each research network is distant from several vantage points, which are often behind interesting configurations (specifically those sites in China and Israel) that can introduce false links and aliases. Further, routers of specific research networks (Abilene, NLR, CANET) do not respond to alias resolution probes, which confounds topology inference.

8. TOPOLOGY ANALYSIS

In this section, we consider the degree distribution and sampling bias apparent in our DisCarte-inferred topology. We chose these properties because they could be affected by the missing or erroneous aliases.

Lakhina et al. [18] introduce a method for evaluating measured network topologies to see sampling bias in the degree distributions of routers. The fundamental assumption is that high-degree routers are equally likely to be anywhere in the topology, and specifically, are no more likely to be near to the sources than farther away. A biased sample would tend to see many of the links incident to nearby routers, because the shortest path tree from a source is more likely to include the links of nearby routers, and less likely to include more than two links on distant routers.

We repeat the analysis of Lakhina et al. and find sampling bias in both DisCarte- and Rocketfuel-inferred topologies. We show the complementary cumulative distribution of the router out-degrees (Figures 9 and 10) in the near set (those within the median distance from a vantage point), in the far set (those of median or greater distance), and overall. That the near set has somewhat higher degree demonstrates sampling bias in the topology. This suggests that more data rather than higher-quality topologies are required to remove bias.

We expect sampling bias to be present in the topology we measure. The best approach to eliminating such a bias is most likely to wildly increase the number of vantage points relative to the destinations as performed in Rocketfuel [40]. Even when doing so, sampling bias is not eliminated: Lakhina's test found bias in all studied topologies.

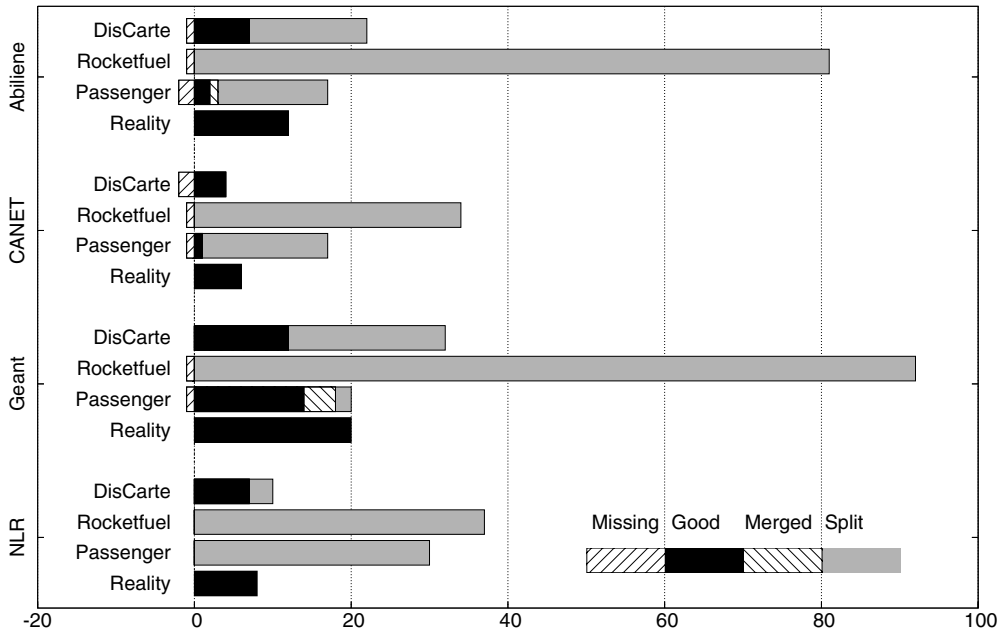

Figure 7: Number of discovered routers (partitioned by accuracy classification) compared to published topologies.

9. RELATED WORK

We classify related work into four categories: network mapping techniques, measurement-based inferences, error-avoidance in traceroute, and error characterization in network maps.

9.1 Internet Mapping

Techniques for Internet mapping present various methods for selecting traceroute measurements and resolving aliases. In 1995, Pansiot and Grad [29] pioneered network mapping by tracerouting to approximately 3000 destinations and introduced alias resolution by source address. Mercator [12] revolutionized mapping through source-routed probes, alias discovery by source routing, and validation against real-world networks CalREN and Los Nettos. Rocketfuel [40] sought fidelity of ten ISP maps by exploiting traceroute servers and added alias resolution by IP address. Skitter [7] and iPlane [21] apply many of these techniques continuously, making a current reference topology available to researchers.

DisCarte is comparable to these projects in that it introduces a new and more complete technique for improving the correctness of the network mapping and a novel method for alias resolution. We approached correctness in the measured topology by measuring each path using two methods (RR and TR) so that we can detect and remove disputed conclusions. These features are crucial to continued network instrumentation because (a) security concerns cause administrators to filter traffic destined for routers and (b) the scale of the network demands such a large scale measurement that some collected traces are certain to have errors, and unlike in the natural sciences, these errors are not averaged out by further measurements.

9.2 Learning and Inference Techniques

Techniques to interpret raw network measurements are sometimes required; these often involve learning techniques to manage the scale of the problem. Padmanabhan used Gibbs sampling and Bayesian learning to discover lossy links [28]. Mahajan et al. [23] used linear constraints to model intra-domain link weights: a study that could imply a means of detecting and removing false links (those that have too high a cost to be used). Yao et al. [45] present a technique for merging anonymous routers—those that do not respond to traceroute that might otherwise be ignored in a topology (potentially partitioning the network) or thought unique on each observation (wildly inflating the path diversity). Finally, Gunes and Sarac [14] use the addressing structure of the network to deduce the prefixes to which interface addresses belong and infer aliases.

9.3 Traceroute Error Avoidance

Although we apply the paired measurement of TR and RR to bolster uncertain measurements, an alternate, but complementary, approach is to reduce the likelihood of error in the first place. Augustin et al. [3] observe that router load balancing is typically flow based: to restrict a traceroute to a single path requires only redesigning the tool to preserve the five-tuple of protocol with source and destination address and port.

Our previous work showed that RR had the potential to detect route changes and could be applied toward a more reliable traceroute [38]. However, we found that simple methods were intractable and were unable to process almost 40% of our data.

9.4 Network Map Errors

Some errors in network maps may be avoided through improved techniques. Teixeira et al. [44] noted a lack of fidelity in Rocketfuel maps: measurements were incomplete (backup links were missing) and aliases were missing or erroneous (some addresses were split and merged). In estimating path diversity, these factors somewhat canceled each other, but the result was not reliable.

10. RECORD ROUTE REDESIGN

If the record route IP option were designed today, it would benefit from more precise standardization and the ability to sample paths longer than 9 hops.

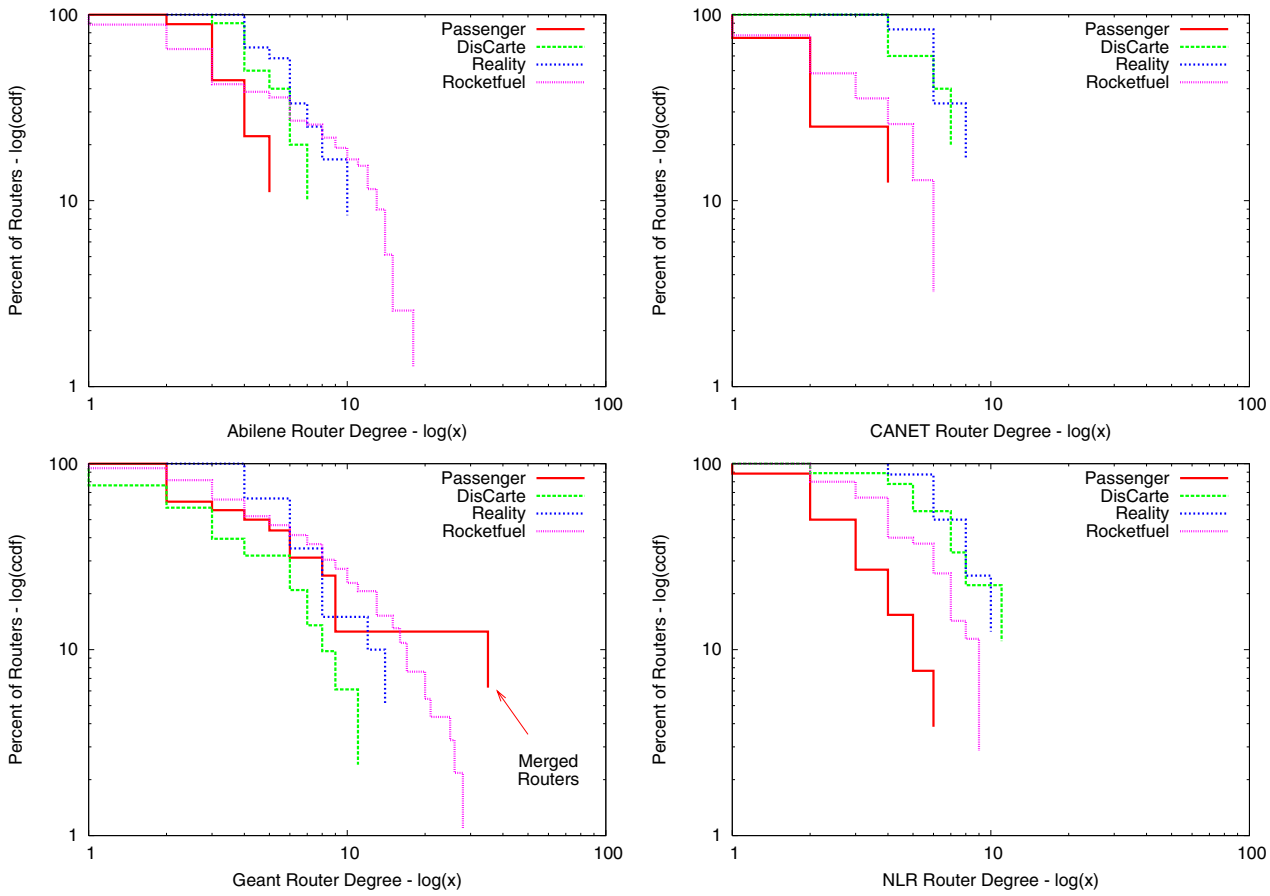

Figure 8: Degree distribution by inference technique: DisCarte-inferred topologies best reflect reality.

Address alignment would be trivial if record route implementations were standardized (and such standards were adhered to). We believe the implementation diversity in record route (Section 3) exists because RFC 791 does not specify how to treat options on expiring packets. For topology discovery, Arriving is the most appealing RR implementation, where addresses are recorded for expiring packets. If this scheme were universal, an alias could be discovered with a single packet.

A more powerful record-route option would include the ability to "skip" a configurable number of addresses before starting to record. In this way, successive RR probes could record 9 hop subsections of a path, giving complete RR information from end to end, as opposed to the current 9 hop limit. Implementation is simple: routers need only increment the RR array index pointer even if the RR array is full, allowing the index to wrap. Thus, the sender sets the initial RR pointer value to $4 - (4 \times k) \bmod 256$ to skip k hops before starting to record the route. Recall that a router along the path only records the route if the pointer value p is in the range $4 \leq p < l$ where l is the length of the RR option in the IP header.

11. CONCLUSION

Internet topology measurement faces a continuing problem of scale: more nodes and links are added, measurement platforms like PlanetLab grow, and filtering policies and implementations remain diverse. To capture this topology requires not simply the ability to collect, store, and query against the 1.3 billion response packets in our data set, but also the ability to filter this data to discern which observations and interpretations are valid. Toward this goal, we adopted disjunctive logic programming to merge our expectations of network engineering practice—common vendor choice and address prefix assignment—to interpret and merge our topology data.

DisCarte provides a novel cross-validation tool for network topology discovery—it finds aliases that increasingly cannot be detected by active probing (30% of addresses we found could not be probed), it finds routers that do not decrement TTL (329) or generate ICMP errors (2,440), it verifies that probed paths are consistent during a measurement—but extracting this information requires significant effort. Expectations of network engineering practice provide the hints required to interpret this data accurately, and a divide-and-conquer approach allows the flexible interpretation to take place quickly over subsets of the data and resolve contradictions.

Our effort owes its inspiration to Vern Paxson's *Strategies for Sound Internet Measurement* [31]. Our approach that led to DisCarte is to measure the same path and topology using two different methods so that their consistency can ensure an accurate result (one of Paxson's "calibration" strategies). Along the way, we adopted many of his hints: study small components first (the PlanetLab topology before the Internet; small cliques before larger ones), invest in visualization (we used neato and dot [13] to compare topologies), build test suites (our regression tests include 77 difficult-to-interpret traces and groups of traces), and we make available our data and analysis scripts.

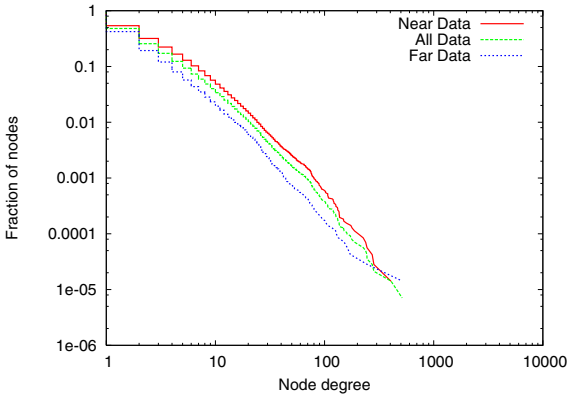

Figure 9: Bias in DisCarte-computed topology.

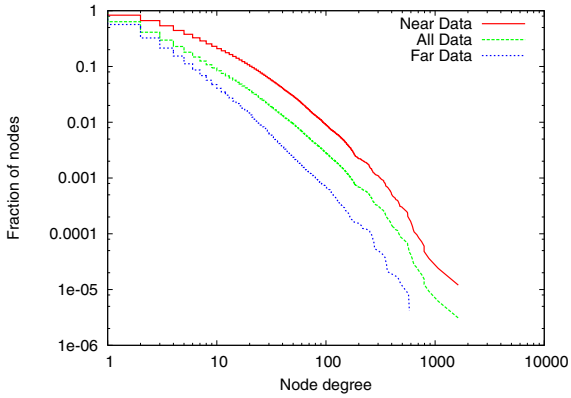

Figure 10: Bias in Rocketfuel-computed topology.

Our future work is to develop two related components: an application-specific version of our (inefficient but general-purpose) DLP-based solver, and a more efficient measurement interpretation scheduler that would choose to study related measurements together to reduce the computational requirements of the analysis. In this first application of record route in topology measurement, getting the right answer took precedence over performance; making the measurements and analysis efficient enough to be repeated will take engineering.

Acknowledgments

We would like to thank the systems administrators at University of British Columbia and Vrije University of Amsterdam for their help in mapping unexplained routing behavior back to their manufacturer. We would also like to thank Bobby Bhattacharjee, Katrina LaCurts, David Levin, Justin McCann, Kevin McGehee, and the anonymous reviewers for their helpful comments.

12. REFERENCES

[1] Abilene router configurations. http://pea.grnoc.iu.edu/Abilene.
[2] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *SOSP*, 2001.
[3] B. Augustin, *et al.* Avoiding traceroute anomalies with Paris traceroute. In *IMC*, 2006.
[4] R. P. Bonica, D.-H. Gan, and D. C. Tappan. ICMP extensions for multiprotocol label switching. Internet Draft (work in progress): draft-ietf-mpls-icmp-05, 2006.
[5] F. Calimeri, W. Faber, N. Leone, and G. Pfeifer. Pruning operators for disjunctive logic programming systems. *Fundamenta Informaticae*, 71(2-3):183–214, 2006.
[6] Personal e-mail from Cisco engineers.
[7] k. claffy, T. E. Monk, and D. McRobb. Internet tomography. *Nature, Web Matters*, 1999. http://www.nature.com/nature/webmatters/tomog/tomog.html.
[8] http://www.dbai.tuwien.ac.at/proj/dlv/examples/3col.
[9] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the Internet topology. In *ACM SIGCOMM*, 1999.
[10] L. Gao and F. Wang. The extent of AS path inflation by routing policies. In *IEEE GLOBECOM*, vol. 3, 2002.
[11] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
[12] R. Govindan and H. Tangmunarunkit. Heuristics for Internet map discovery. In *INFOCOM*, 2000.
[13] Graphviz. http://www.graphviz.org.
[14] M. H. Gunes and K. Sarac. Analytical IP alias resolution. In *IEEE International Conference on Communications (ICC)*, 2006.
[15] N. Hu, O. Spatscheck, J. Wang, and P. Steenkiste. Locating Internet bottlenecks: Algorithms, measurements, and implications. In *ACM SIGCOMM*, 2004.
[16] C. Jin, Q. Chen, and S. Jamin. Inet: Internet topology generator. Tech. Rep. CSE-TR-433-00, University of Michigan, EECS dept., 2000. http://topology.eecs.umich.edu/inet/inet-2.0.pdf.
[17] E. Katz-Bassett, *et al.* Towards IP geolocation using delay and topology measurements. In *IMC*, 2006.
[18] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling biases in IP topology measurements. In *INFOCOM*, 2003.
[19] N. Leone, *et al.* The DLV system for knowledge representation and reasoning. *ACM Trans. Computational Logic*, 7(3):499–562, 2006.
[20] M. Litzkow, M. Livny, and M. Mutka. Condor: A hunter of idle workstations. In *ICDCS*, 1988.
[21] H. V. Madhyastha, *et al.* iPlane: An information plane for distributed services. In *OSDI*, 2006.
[22] P. Mahadevan, D. Kriokov, K. Fall, and A. Vahdat. Systematic topology analysis and generation using degree correlations. In *SIGCOMM*, 2006.
[23] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. Inferring link weights using end-to-end measurements. In *IMW*, 2002.
[24] Z. M. Mao, J. Rexford, J. Wang, and R. Katz. Towards an accurate AS-level traceroute tool. In *ACM SIGCOMM*, 2003.
[25] A. Medina, I. Matta, and J. Byers. BRITE: A flexible generator of Internet toplogies. Tech. Rep. BU-CS-TR-2000-005, Boston University, 2000.
[26] D. Meyer. University of Oregon Route Views project. http://www.routeviews.org/.
[27] A. Nakao, L. Peterson, and A. Bavier. A routing underlay for overlay networks. In *ACM SIGCOMM*, 2003.
[28] V. N. Padmanabhan, L. Qiu, and H. J. Wang. Passive network tomography using Bayesian inference. In *IMW*, 2002.
[29] J.-J. Pansiot and D. Grad. On routes and multicast trees in the Internet. *ACM CCR*, 28(1):41–50, 1998.
[30] V. Paxson. End-to-end routing behavior in the Internet. *IEEE/ACM Transactions on Networking*, 5(5):601–615, 1997.
[31] V. Paxson. Strategies for sound Internet measurement. In *IMC*, 2004.
[32] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the Internet. In *HotNets*, 2002.
[33] J. Postel, editor. Internet protocol. IETF RFC-791, 1981.
[34] F. Ricca, W. Faber, and N. Leone. A backjumping technique for disjunctive logic programming. *The European Journal on Artificial Intelligence*, 19(2):155–172, 2006.
[35] E. C. Rosen, A. Viswanathan, and R. Callon. Multiprotocol label switching architecture. IETF RFC-3031, 2001.
[36] S. Savage, *et al.* The end-to-end effects of Internet path selection. In *ACM SIGCOMM*, 1999.
[37] R. Sherwood and N. Spring. A platform for unobtrusive measurement on PlanetLab. In *USENIX Workshop on Real, Large Distributed Systems (WORLDS)*, 2006.
[38] R. Sherwood and N. Spring. Touring the Internet in a TCP sidecar. In *IMC*, 2006.
[39] N. Spring, R. Mahajan, and T. Anderson. Quantifying the causes of path inflation. In *ACM SIGCOMM*, 2003.
[40] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with Rocketfuel. In *ACM SIGCOMM*, 2002.
[41] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: A public Internet measurement facility. In *USITS*, 2003.
[42] J. Strauss, D. Kitabi, and F. Kaashoek. A Measurement Study of Available Bandwidth Estimation Tools. In *IMC*, 2003.
[43] H. Tangmunarunkit, R. Govindan, and S. Shenker. Internet path inflation due to policy routing. In *SPIE ITCOM Workshop on Scalability and Traffic Control in IP Networks*, vol. 4526, 2001.
[44] R. Teixeira, K. Marzullo, S. Savage, and G. Voelker. In search of path diversity in ISP networks. In *IMC*, 2003.
[45] B. Yao, R. Viswanathan, F. Chang, and D. Waddington. Topology inference in the presence of anonymous routers. In *INFOCOM*, 2003.
[46] E. W. Zegura, K. Calvert, and S. Bhattacharjee. How to model an internetwork. In *INFOCOM*, 1996.