Topic: Topology and Backbone Networking
DisCarte: A Disjunctive Internet Cartographer

Sebastian Schlunke
(schlunki@cs.tu-berlin.de)

Seminar "Internet Measurement"
Technische Universität Berlin

WS 2008/2009 (version of January 27, 2009)

Abstract

This paper covers problems of current topology inference techniques and introduces DisCarte, a topology inference tool, that uses the IP record route option and disjunctive logic programming to augment existing techniques. It is based on the paper "DisCarte: A Disjunctive Internet Cartographer" by R. Sherwood, A. Bender and N. Spring [Dis08].

Topologies that are solely based on traceroutes and direct probing are often inaccurate, due to unresponsive routers and network architectures, that have grown in size and complexity. The record route option provides independent data, that can be used to verify traceroute data and enhance topologies. DisCarte uses disjunctive logic programming to combine traceroute and record route data. With the aid of observed network engineering practices results will be generated, that are closer to reality.

I will discuss what the problems of current techniques are and how they are addressed by DisCarte. A divide and conquer approach will be presented, that allows to scale the process to Internet size networks and it will be evaluated, to what extent Discarte produces better results than other techniques.

1 Introduction

Information about the topology of a network, its internal structure, is of great interest to researchers, developers and network operators. Such knowledge can be used to "determine where losses, bottlenecks, failures and other undesirable and anomalous events occur". Also topologies, if precise enough, "benefit overlay constructions, network diagnostics, modeling and measurement"[Dis08].

Yet, this information is for various reasons rarely published by the network's operator and the IP protocol does not provide support that would allow a direct analysis of the network structure [Dis08]. The network topology has to be deduced from data gained by measurements and observations. Tools currently used to infer network topologies are, for example, Rocketfuel, Paris traceroute or Passenger [Par06, Pas06, Rock02]. To gather the necessary data, all of them heavily depend on basic techniques like TTL-limited-probes, i.e. traceroute and direct router probing. These methods are known to produce inaccurate or erroneous data under certain circumstances. In order to increase

the completeness and correctness of discovered topologies it is required, that these errors or inaccuracies are detected and, if possible, corrected.

"The often-ignored record route (RR) IP option provides a source of disparate topology data"[Dis08] that can be used to complete and correct the data collected by traceroute. The Passenger tool was developed using this approach and it has been observed, that the different implementations of the record route option in routers pose a significant problem when aligning traceroute and record route data [Pas06].

DisCarte addresses this problem. By using disjunctive logic programming (DLP), traceroute and record route (RR) data is intelligently merged and crossvalidated "against observed network engineering practices"[Dis08], providing more accurate and complete results, than previous techniques.

In the following section traceroute based topology discovering techniques, their problems and limits, will be discussed before the RR option is introduced. How current techniques can be augmented by using RR will be shown in Section 3. In Section 4 DisCarte will be presented, its methods examined and in Section 5 the results will be discussed and compared to conventional tools.

2 Background

In this section topology discovering based on traceroute is shortly described. It will be discussed why this is error-prone, what circumstances cause failures and how these failures distort the inferred topology. Afterwards the technical details of the RR option and the resulting limits are presented.

2.1 Problems of Traceroute-based Topology Discovering Techniques

A router-level network topology consists of two things which have to be inferred, links and aliases. While a link represents a connection between two IP addresses of different routers, an alias pools different IP addresses that belong to different interfaces of the same router. Links are found by traceroutes, whereas aliases can identified by direct router probing.

Some sources of error are inherent to these techniques. Traceroute identifies links, by sending packets with ascending TTL values until the target is reached. Each of these packets should be answered by an ICMP time-exceeded reply. This answer contains the IP address of the incoming interface of the corresponding router on the path to the target. The problem is, that "TR [traceroute] assumes that sequential probes traverse the same paths"[Dis08] and therefore concludes, that two routers are linked, if their IP addresses have been exposed by ICMP replies to packets, whose TTL value differed by one.

But sequential probes are not necessarily routed on the same path, e.g. due to load balancing. These "mid-measurement path instabilities cause TR [traceroute] to infer incorrect links"[Dis08], because links, that do not exist are included in the topology, while links that do exist, are missed. Routers may also be missed, if all traceroute probes with appropriate TTL expire on alternative paths.

The results of a traceroute are further distorted by anonymous routers, hidden routers and routers, that do not respond to direct probing.

Anonymous routers do not send an ICMP reply when packets expire, causing missing information that forms a significant source of error as links cannot be assigned and aliases not resolved [Dis08].

Hidden routers do not alter the TTL value of a forwarded packet. As a consequence they never generate an ICMP reply and cannot be detected by traceroute-based topology discovery. This results in missing nodes and incorrect link inferences.

Routers, that do not respond to direct probing ignore packets directly addressed to them, thus making alias resolution techniques[1] based on characteristics of the IP packet or the source address impossible. Unresolved aliases are biasing the degree distribution of the network because they increase the number of detected routers in the topology, while the number of links that each router has is decreased.

This behavior can be seen in Figure 1. The topology derived by Rocketfuel shows a lot more routers than actually exist. Aliases have not been resolved and each inferred router has only one interface. DisCarte however was able to resolve some aliases, inferring a topology that is closer to reality. In [Dis08] it was observed, that a significant number of routers (32%) do not respond to direct probing, making this a significant source of inaccuracy.

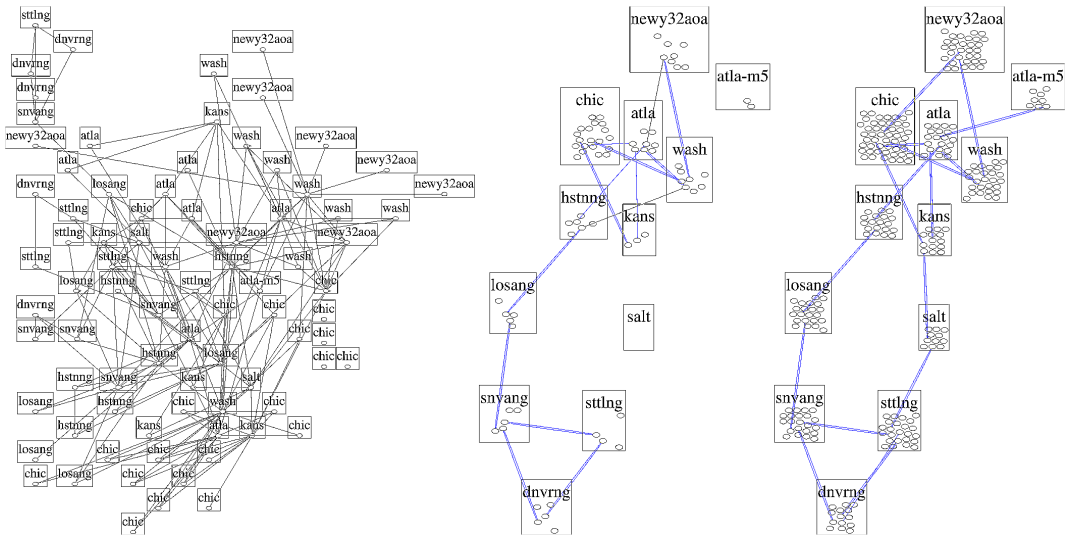

Figure 1: Abilene topology: inferred by Rocketfuel (left), DisCarte (middle), and actual topology (right), rectangles are routers with interior ovals representing interfaces (taken from [Dis08])

2.2 The Record Route Option

In addition to vital information like source and destination address IP datagrams (as defined in [2]) provide a set of options, which can be used by the sender of an IP packet to manipulate how a packet is processed by its receiver. One of these options is Record Route. Within certain limitations, this option allows the sender to track the "path" a packet takes. In the next sections the technical details of the RR implementation are described and resulting problems discussed.

2.2.1 Technical Background

The options for an IP packet are stored in its header. The header, which can be described as an array of bytes, can contain an area for options at its end. The RR option uses a part of this area to store the addresses of the routers which forwarded the packet.

[1] A detailed description of these techniques can be found in [Rock02].

A router that forwards a packet with the RR option is instructed to add "its own Internet address as known in the environment into which this datagram is being forwarded into"[2] to the packets list of addresses.

Although the RR option allows to specify how much space is to be used for the address list, the IP header itself is limited in size. It can be at maximum $15 \cdot 4$ byte blocks long, of which 5 are reserved by definition and 1 is needed for the option itself, which leaves $9 \cdot 4$ byte blocks for the address field. A packet whose address array is already full, shall be forwarded without modifying the addresslist [2].

Therefore up to 9 addresses of the path a packet takes, can be recorded by the RR option.

It should also be noted, that in the case of a TTL expiration, the header of the expired packet is returned in the ICMP reply, including the RR option.

2.2.2 Limits & Problems

Due to its nature, the RR option is restricted in its usefulness and can even cause problems.

The RR option can only add information of the first nine hops on the path, independent from the target. For larger networks, especially the Internet, nine hops is a very small number. Thus the data that can be added with RR is strictly limited by the availability of a "geographically diverse set of vantage points"[Dis08], so that as much hosts as possible can be reached within nine hops. Only for these parts of the network, RR can increase the accuracy and completeness of the derived topology.

When using IP options like RR, it should be considered, that they are rarely used. Packets with IP options set may be dropped by Firewalls, in which case just no information is obtained, or even trigger intrusion detection systems, which target anomalous events. Therefore probes with the RR option set should be done cautiously to prevent reports of abuse.

3 Use of Record Route in DisCarte

DisCarte uses the RR option to improve the results of the analysis. The information gathered with RR is merged with the data collected by conventional traceroutes in a process called *address alignment*.

It will be discussed, how information is gathered by RR and what the limits and problems are. Also the difficulties, which have to be considered when addresses are aligned, will be studied and the benefits of using RR shown.

3.1 Address Alignment

In order to use the data gathered with RR, it has to be merged with the traceroute information. Every IP address discovered by traceroute has to be associated with an IP discovered by RR and vice versa. This seems to be an easy task, as the router belonging to an entry in the address list should be identifiable by its TTL distance to the source of the packet.

Unfortunately several problems arise when collecting RR data, complicating the aligning of addresses.

3.1.1 Different Implementations

Although the RFC 791 clearly states how routers should behave, different kinds of RR implementations have been observed. The observed implementations are:

Departing: The router updates the RR array with the address of its outgoing interface when the packet is forwarded and leaving the router. If the packet expires no RR entry is made.

MPLS: RR arrays are updated like in Departing routers, except for packets that arrive over an interface, that is MPLS[2]-enabled. In this case the RR array is not modified.

NotImpl: The router ignores the RR option and does not modify the RR entries.

Arriving: The RR array is modified, when the packet arrives. But the address used can be either the one of the outgoing interface or the internal loopback device.

Lazy: These routers update the RR array for packets with the RR option set, but they do not decrement the TTL.

Mixed: Mixed routers behave like Departing routers, if the packet does not expire. If the packet expires, the RR array is updated with the address of the incoming interface.

To correctly align addresses, it is necessary to assign an implementation type to each router in a trace. The diversity of the implementations complicates the alignment process, as an assignment is not necessarily distinct. Especially the NotImpl and Hidden types cause ambiguous assignments for a given trace and RR array. We will see in Section 4.2.2 how this problem is handled by DisCarte.

3.1.2 Topology Traps

Topology discovery can be complicated by certain characteristics of the very topology that is to be discovered. Like the RR implementation in a router these "topology traps"[Dis08] have to be detected and considered when merging data.

Hidden routers: This rare type of router cannot be discovered by the means of traceroutes. But they may be detected by RR. This has to be taken into account by the inference tool.

Non-standard firewall policies: Packets with RR option set may be treated differently than those without. This can result in working traceroutes without RR, but may cause dropped packets or useless replies with RR.

Enabling IP options breaks load-balancing: IP packets with RR option set seem to break some load-balancing algorithms, which would normally send packets belonging to the same data flow to the same path. Packets with IP options may be send over arbitrary paths of equal costs.

Different-length equal-cost paths: These paths can produce probes which have different distances between the same source and destination. These probes may only be compared if they traversed the same path or they "may cause false topology assertions" and "create false links and aliases"[Dis08]. The RR data can be used to divide the probes properly so that wrong conclusions are avoided.

RR fills: Due to the different RR implementations in routers, one hop may add more than one RR entry. If the packet already contains eight entries, only one entry can be made and the information for the second entry and the true number of entries are lost. "For example, a packet with eight RR entries that transitions from a Departing RR-type router to an Arriving RR-type router would normally receive two new RR entries."[Dis08]

[2]Multiprotocol Label Switching: a technique that allows routers to forward connection-oriented packets [1]

3.2 Advantages when using RR

The use of the RR IP option has several advantages. The first thing to be considered is, that RR may be combined with traceroute, collection RR and traceroute data simultaneously. As explained in Section 2.2.2 the RR option may lead to unexpected behavior in some cases, limiting the data that can be gathered with combined traceroutes. Another possibility is to run the traceroute twice, with and without RR. In this case RR "provides a source of disparate topology data"[Dis08], which means, that any information gained with RR can only improve results that are based on traceroutes alone.

If address alignment is successful, RR can compensate some of the deficiencies of traceroute, producing a more accurate and complete result.

It is possible to:

- resolve aliases without direct probing,
- expose hidden and anonymous routers (see Figure 2),
- discover multi-path load balancing and expose mid-measurement path instabilities.

If a router implements the RR option, the IP address of one interface is revealed each time a packet traverses the router. Aliases can be resolved, when packages traverse the router in multiple directions. By using different incoming and outgoing interfaces, each time another address of the router is discovered. However, the exact RR implementation of a router must be known to assign all the detected addresses correctly to that router.

Hidden and anonymous routers will be found, if they implement the RR option, using their entry in the address list. This requires, that the other routers on a path containing the hidden/anonymous router have been correctly aligned, so that the entry belonging to the router in question can be correctly identified.

As the RR option allows to track the path of a packet, its possible to decide, wether two packets traversed the same path. In the case of path instabilities and load balancing the track record of affected packets would differ among them. By separating those probes, false conclusions, that would have been possible when using traceroute data alone, can be prevented when inferring the topology.

Figure 2 is an example, that compares a Rocketfuel inferred topology, which uses only the means of traceroute, to a topology derived by DisCarte, that combines traceroutes and RR. It can be seen, that DisCarte finds an anonymous router and load balanced paths, whereas Rocketfuel misses both. The load balanced paths use different inferfaces of the anonymous router R3. The missing router is indicated by the dotted rectangle in the lower topology.

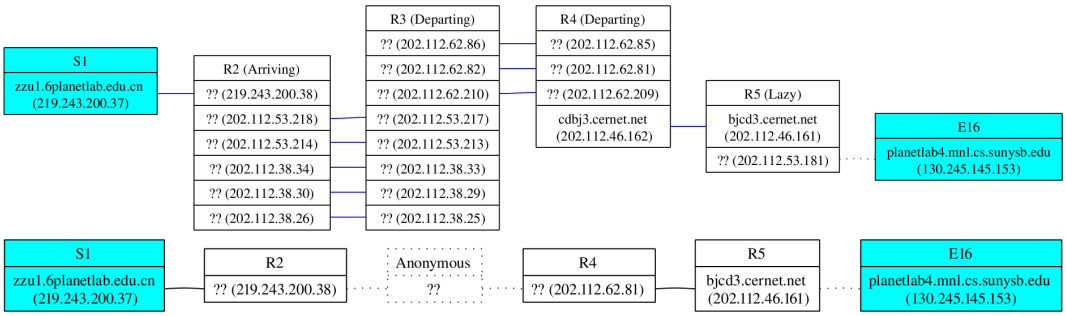

Figure 2: Partial Trace from Zhengzhou University, China to SUNY Stony Brook, USA; inferred by DisCarte (top) and Rocketfuel techniques (bottom). DisCarte finds many load-balanced path through an anonymous router (R3) (taken from [Dis08])

4 Inferring Topologies with DisCarte

With DisCarte a new approach in how to combine TTL-based data with RR-based data has been presented. As explained in the previous sections, topologies inferred exclusively from traceroutes can be improved by information gathered with RR, but only if both data sets can be merged, that means that the found addresses have to be correctly aligned. An earlier approach to this was made with Passenger, which used heuristics to align data, but it showed that due to the limited flexibility of the heuristics only parts of the data could be aligned [Pas06].

To align addresses and validate the result of the process, DisCarte utilizes disjunctive logic programming (DLP) [Dis08]. In this chapter, a brief introduction to DLP will be given and it will be shown how it is used by DisCarte to infer topologies. Subsequently the crossvalidation and scaling techniques are presented, before the quality of the results is discussed.

4.1 Disjunctive Logic Programming

DLP is an area within constraint logic programming [3]. A logic program generally consists of facts and rules. Facts are statements that are, in terms of the logical program, given and always true. Rules somehow combine facts and rules with (mostly) logical operators. Within disjunctive logic programming rules may look like:

$$\text{fact}_0 \Rightarrow \text{rule}_1 \vee \cdots \vee \text{rule}_n$$

Thus the validity of one rule must be deducible from a fact. This fits the needs of DisCarte, as exploration with traceroute produces a lot of information, which will be converted to facts.

The so-called solver will try to find possible assignments for variables in a rule, so that it becomes true. DisCarte will use this, to find possible RR implementations for routers.

Additionally the DLP implementation used by DisCarte allows to define constraints on the solutions and to price the solutions. There are strong and weak constraints. A model that violates a strong constraint, can never be a solution to a logical program. Weak constraints raise the costs of a model, if they are violated. "The output from a DLP is the lowest cost model of inferred facts generated from input facts and inference rules."[Dis08]

4.2 Topology Generation with DLP

In order to generate a logical program, two things have to be supplied to the DLP solver by DisCarte: facts and rules.

Facts will be generated from the raw trace data. They combine traceroute and RR data in so-called probe pairs. The rules have been written as part of DisCarte. As earlier mentioned, they have to consider the different RR implementations in the routers and the topology traps.

4.2.1 DLP Fact Generation

The facts, that will be used in the DLP solver "consist of both straightforward parsing of the data, deriving facts more easily computed without DLP, and *probe pairs*"[Dis08].

Facts that can be obtained by simple static analysis of the input data, are for example routers with Mixed or Lazy RR implementation. The Mixed RR implementation is the only type, that can cause a router to insert its incoming interface address into the RR

array. If a packet expires at router X with the appropriate ICMP reply and the last RR entry is also X, then that router has to be of the Mixed RR type. This type can be detected easily and added as fact without further concern.

This is different for the Lazy RR type. Routers with Lazy RR implementation do not alter the TTL value, if IP options are set. That means, that packets with RR option will never expire at such a router X, but at the next one, router Y. This makes it possible, to identify Lazy routers. A router is Lazy, if all non RR probes with TTL t expire at router X and all non RR probes with TTL $t+1$ expire at Router Y, but RR probes with TTL t expire at router Y. These wrong distances have to be corrected, before probe pairs are formed.

The probe pairs are assembled from two subsequent TTL-limited probes. They have the following form: "$probePair(p_1, p_2, \delta)$, where p_1 and p_2 are unique probe identifiers, and δ is the difference between the size of the two RR arrays"[Dis08].

This is trivial for traceroute-only data, but may include errors, e.g. mid-measurement path instabilities as discussed earlier, that go unnoticed.

The RR information can now be used, to make probe pair identification more accurate. Probes, that have not traversed the same path, can be identified as the RR record tracks their path. If the two probes do not share the same path, except maybe for the last entry of the probe that went one hop further, they must not form probe pairs. Last, if Lazy routers lie on different length paths, it is only possible to identify them on the path used by non-RR probes [Dis08]. Information on other paths has to be discarded.

4.2.2 DLP Rules

The function of the DLP rules is to assign a implementation type to each router. Therefore they describe all implementation types that would be possible for two routers in a given probe pair and given δ.

A rule consists of a disjunction of terms in the form of:

$$transition(X, Y, \text{RR-Type}_X, \ldots, \text{RR-Type}_Y)$$

Such a transition rule either "assigns" a RR type to a router if X or Y are variables, or tests wether X and Y have the required RR type. In this example, router X either has or becomes assigned the RR implementation RR-Type$_X$, router Y respectively. As there are routers, that are not found by solely traceroute-based techniques, like Hidden routers, there may be nodes between X and Y, not affecting the TTL but increasing the number of RR entries as well as δ.

To form a rule, these disjunctions are combined with a probe pair and a constraint to δ, e.g.:

$$\Leftarrow probePair(X, Y, \delta), \delta = 1.$$

As mentioned, probe pairs were defined as facts, with identifiers for every found router. Such a rule demands, that if the facts contain a probe pair with $\delta = 1$, then the routers X and Y either already have a RR type so that one of the transition rules in the implication holds true, or they can be assigned one.

An example for a rule used in DisCarte (taken from [Dis08]):

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

This rule accumulates all possibilities for a probe pair whose RR arrays differ in size by one. It should be noted, that also "special" cases, like Hidden routers or ones that do not implement RR, have to be considered, which inflates the number of rules. As routers can only have a distinct RR type, exactly one of the cases above has to apply.

As explained in the previous section, δ measures the difference in size of the RR-arrays of two subsequent probes. Because of Hidden routers δ could be any positive number, but due to the limited size of the RR option, only values in the range from 0 to 9 have to be considered. Furthermore the size of the rules increases dramatically for greater δ and as values greater than 4 have not been observed by the developers of DisCarte [Dis08], the rules cover only the values from 0 to 4.

4.2.3 Generated Solutions

The generated facts and defined rules allow the DLP solver to generate possible RR type assignments for all found routers. As mentioned above, these solutions are pruned by strong constraints, which reduces the number of solutions, and valued by weak constraints, which establishes an order between solutions.

There is only one strong constraint on the solution. All routers in a solution must have the same RR implementation for all their interfaces respectively, or the solution cannot be valid. A single router must not have different RR implementations on different interfaces.

The weak constraints, which if violated raise the costs of a solution, have been chosen to model several observed engineering practices, techniques or configurations that seem to be common and often used. They may be violated in some cases, but if they are too often, it can be assumed, that the model does not reflect reality. "Thus the model that violates the fewest practices is likely to be the closest approximation of reality."[Dis08]

Here is a list of weak constraints as used in DisCarte, ordered descending by their costs:

1. No self-loops: a router should never forward packets directly to itself. It is likely, that in such a solution two distinct routers have been merged due to a wrongly assigned alias.
2. In order to conserve address space by using the smallest network block possible, network architects often assign IP addresses on either side of a link, that differ only in one bit. (An example for this can be seen in Figure 2)
3. If a router supports direct probing, the obtained information about its aliases are often correct and should be used over the inferred ones.
4. Hidden routers have been observed to be rare, thus a solution with few hidden routers is better than a solution that contains many of them.

5. Routers that do not implement RR been observed to be more rare that those who do, thus a solution with few router of RR type NotImpl is better than one with many.

It is possible, that several solutions with the same costs are generated. In this case the information has not been sufficient enough to enforce a unique solution. On the other hand, there may be no solution. This is likely to be the case, if the data was contradictory or the model erroneous.

4.3 Handling Large Networks

The process presented above, has to be adjusted before it can be applied to large, or Internet sized networks. This is because large networks produce a lot of measurement data, i.e., a lot of probe pairs are generated and "the number of possible RR implementation assignments grows exponentially with the number of probe pairs."[Dis08]

DisCarte tries to cope with this problem, by using the divide and conquer pattern. The data is processed in small parts, reducing the solution space for each part considerably. Then the solutions are merged together. When merging the data, conflicts may occur. That means, that in one solution two IPs were deduced to be aliased, while in another solution they appear to be linked. These conflicts have to be solved, before a topology can be extracted.

The data is processed in the following way:

1. All two-cliques, that means for all sites $S1$ and $S2$ the traces from site $S1$ to $S2$ and vice versa, are computed. (see Figure 3)
2. For all triangle-like subsets the traces from $S1$ and $S2$ to all destinations D are computed, as shown in Figure 3. As the paths between $S1$ and $S2$ have already been computed and contain no conflicts anymore, they can be used as overlap for cross-validation when computing the paths to D.
3. The deduced facts are extracted and compared to expose contradictions.
4. In the case of a conflict, the input sets of the conflicting models are processed together via DLP. If this produces a single solution, the conflict is solved and the correct state of the IPs is transferred to all affected models, which are recomputed via DLP. Multiple solutions indicate, that the information is insufficient to solve the conflict, while no solution can hint at erroneous input data or a new RR behavior.

It has shown that this technique provides datasets, that are small enough to be processed rapidly, but do still provide enough information for the DLP to produce meaningful results. Also the number of conflicts when merging the results are reduced. However, if a conflict cannot be resolved, all related facts have to be removed from the model, as their data is not reliable.

5 Results

The results produced by DisCarte have been tested and judged regarding accuracy and completeness. First the ability to deduce aliases has been compared to those of Rocketfuel's alias resolution tool Ally. Second, topologies inferrend by DisCarte, Rocketfuel and Passenger have been compared among each other and against published topologies of four research networks.

In probes between PlanetLab nodes, and from PlanetLab nodes to the advertised BGP prefixes, it showed, that DisCarte found about 11% more aliases than the direct probe-based techniques of Rocketfuel's Ally. Of these aliases 92% had only been found because of the added RR option [Dis08].

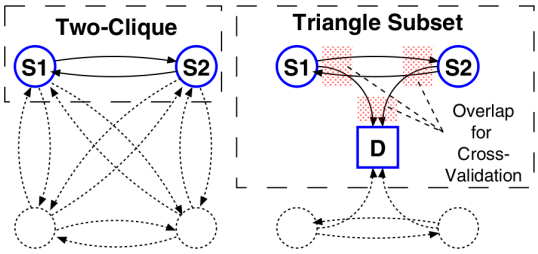

Figure 3: First all addresses in two-cliques (left) between all sources are aligned and then subset triangles (right) to all destinations increasing overlap and decreasing errors. (taken from [Dis08])

The evaluation of the results produced by DisCarte, Rocketfuel and Passenger regarding the inferred routers can be found in Figure 4. The figure shows for each tool the number of correct identified routers (Good), the number of routers that have not been found (Missed), the number of routers that have been merged and mistaken for single router (Merged) and the number of routers, where aliases have been mistaken as distinct router (Split). It can be seen, that Rocketfuel is not able to resolve any aliases, because of routers unresponsive to direct probing. Thus a single router is assumed for each IP, producing a lot of "splitted routers". DisCarte and Passenger in contrast are able to correctly identify some aliases, due to the RR option. It also shows, that DisCarte was in three of four tests able to identify more routers correctly than any other tool and that there were no incorrectly merged routers.

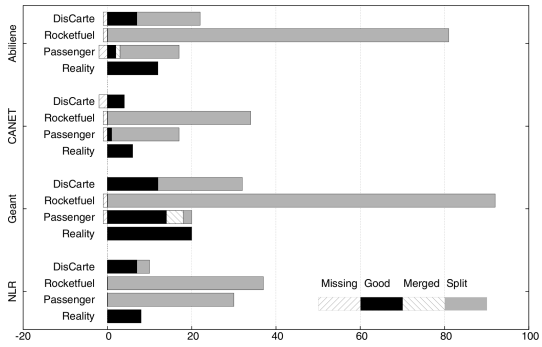

Figure 4: Number of discovered routers compared to published topologies. (taken from [Dis08])

Additionally DisCarte inferred no false links and discovered more than 63% of the existing links [Dis08].

6 Conclusion

Topology discovery is becoming an increasingly difficult task, especially for the Internet. Not only that the networks are steadily growing, in both, size and complexity, but also the data that can be acquired in conventional ways, becomes less informative and connected. And although it is easy to gather a huge amount of data, this is not sufficient enough to create models, that are an accurate image of reality. The ability to reconnect the gathered facts and draw reliable conclusions is required to produce usable results.

DisCarte has proved, that the combination of disjunctive logic programming, the IP record route option and conventional traceroutes significantly increases the quality of the inferred topologies. The record route option improves results in two ways. On the one hand RR is able to verify traceroute data, as it ensures path consistency, and on the other hand it can find aliases and routers, that cannot be detected by traceroutes.

DLP provides a simple yet powerful basis for the modeling of alignment rules and the crossvalidation of inferred data with network engineering practices, allowing a more accurate alignment and interpretation. But the power of DLP comes at the costs of higher computing times. However, the divide and conquer approach allows scaling the process to the mass of data that is generated, when analyzing larger networks.

Therefore if the analyzed networks are small or sufficient processing power is available, then DisCarte is an alternative that produces more exact result than alternative traceroute-based discovery tools.

References

[Dis08] Rob Sherwood, Adam Bender, Neil Spring: *DisCarte: A Disjunctive Internet Cartographer*; ACM SIGCOMM 2008.

[Pas06] Rob Sherwood, Neil Spring: *Touring the Internet in a TCP Sidecar*; ACM 2006.

[Par06] B. Augustin, et al.: *Avoiding traceroute anomalies with Paris traceroute*; ICM 2006.

[Rock02] N. Spring, R. Mahajan, D. Wetherall: *Measuring ISP topologies with Rocketfuel*; ACM 2002.

[1] Wikipedia: `http://de.wikipedia.org/wiki/MPLS`; version of January 18, 2009.

[2] J. Postel, editor. Internet protocol: IETF RFC-791 `http://tools.ietf.org/html/rfc791`; 1981.

[3] Frieder Stolzenburg: *A Flexible System for Constraint Disjunctive Logic Programming*; University of Koblenz 1996.