Palmtree:An IP Alias Resolution Algorithm with Linear Probing Complexity

M. Engin Tozal, Kamil Sarac

engintozal,ksarac@{utdallas.edu}
The University of Texas at Dallas, Dept. of Computer Science Richardson, TX 75080 USA

Abstract

Internet topology mapping studies utilize large scale topology maps to analyze various characteristics of the Internet. IP alias resolution, the task of mapping IP addresses to their corresponding routers, is an important task in building such topology maps. In this paper, we present a new probe-based IP alias resolution tool called `palmtree`. `Palmtree` can be used to complement the existing schemes in improving the overall success of alias resolution process during topology map construction. In addition, `palmtree` incurs a linear probing overhead to identify IP aliases. The experimental results obtained over Internet2 and GEANT networks as well as four major Internet Service Providers (ISPs) present quite promising results on the utility of `palmtree` in obtaining more accurate network topology maps.

Keywords: Internet, topology, map, alias, router

1. Introduction

Internet consists of many networks operated by different organizations that do not necessarily publish their complete network topology maps. On the other hand, having an accurate topology map of the network helps us to understand its structural and operational characteristics such as reliability, connectivity, robustness, and efficiency; enhance current and future protocols; optimize networking structure; and improve synthetic network graph generators and other simulation tools.

Many successful research projects and efforts attempting to derive an accurate and large scale topology map of the Internet have been introduced in the literature [14, 16, 20, 13]. These efforts focus on different but correlated topology maps: *IP level* maps show IP addresses that are in use on the Internet; *router level* maps group the interfaces hosted by the same router into single units; *subnet level* maps enrich the router level maps with subnet level connectivity info; and *AS level* maps demonstrate the adjacency relationship between Autonomous Systems (ASes).

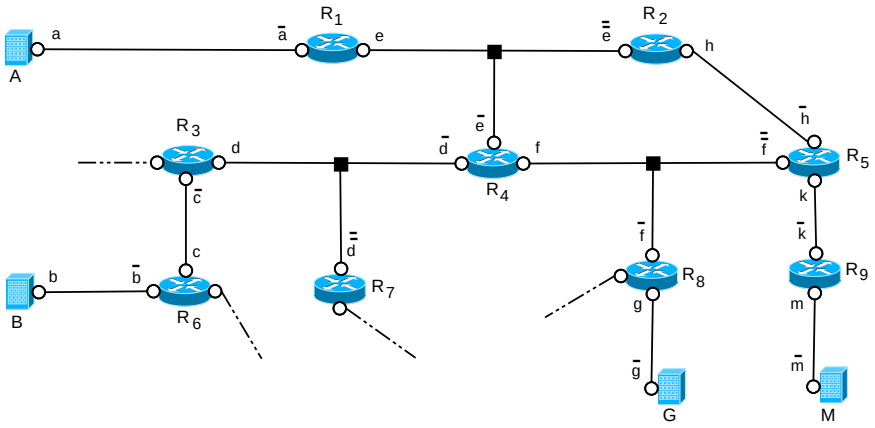
Obtaining router level topology maps in the Internet includes two steps: (1) collecting topology data from the target network and (2) processing this data to build a representative map corresponding to the target network. The common practice in data collection is to use the well-known `traceroute` [10, 1] tool to collect path traces crossing over the target network. These traces are run from a set of vantage points towards a

set of destinations such that each trace crosses over the target network and returns a partial topology of the underlying target network. Processing the collected data involves several tasks including subnet inference [7, 18], star elimination [8], and IP alias resolution [4, 19, 5, 3, 18]. *IP alias resolution*, the task of identifying IP addresses that are accommodated by the same router in the target network, is the main focus of the work presented in this paper. IP alias resolution is an important task and inaccuracies in this task may significantly affect the accuracy of the resulting map with respect to the actual topology of the target network [6].

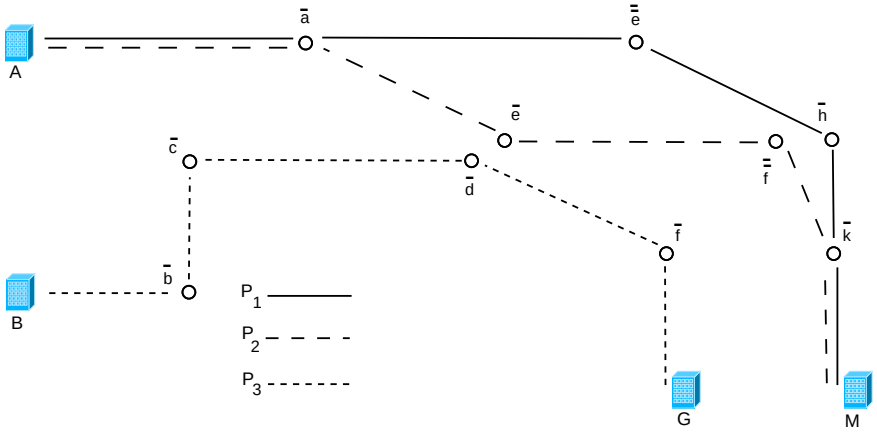
As an example, consider the use of `traceroute` in designing resilient overlay network systems where the goal is to use the obtained traces to identify node disjoint overlay paths. Figure 1a shows the physical topology of a network that includes several routers as well as point-to-point and multi-access links. In the figure routers and hosts are represented with ovals and boxes, respectively and they are labeled with upper case letters. Interfaces are shown with small circles attached to routers and named with lower case letters. Additionally, we used bars (e.g., $\overline{i}$) to keep the picture lucid and to discretely label the interfaces (or IP addresses) being hosted on the same subnet instead of using more lowercase letters. Assume that our goal is to identify node disjoint paths between A and M and between B and G in this network. To keep it simple, let us say that the routers in the figure are responsive to `traceroute` probe packets and they report back the IP address of the incoming interface through which the packet has reached to the router. Figure 1b shows the network topology collected by `traceroute` without using IP alias resolution where $P_1 = \{A, \overline{a}, \overline{\overline{e}}, \overline{h}, \overline{k}, M\}$ and $P_2 = \{A, \overline{a}, \overline{e}, \overline{\overline{f}}, \overline{k}, M\}$ are the two shortest paths between A and M and $P_3 = \{B, \overline{b}, \overline{c}, \overline{d}, \overline{f}, G\}$ is the only shortest path between B and G without IP alias resolution. Based on this topology map one would infer that the use of P_2 for A to M path along with the use of P_3 for B to G path would satisfy the node disjointness requirement. Yet, this would be an inaccurate conclusion as the second IP address ($\overline{e}$) of path P_2 and the third IP address ($\overline{d}$) of path P_3 belong to the same router R_4. However, without IP alias resolution this fact is hidden in the topology map in Figure 1b. As a result, these two paths are not really node disjoint. On the other hand, Figure 1c showing the `traceroute` paths after IP alias resolution would reveal the fact that those two paths share a common router R_4. Hence, IP alias resolution would help to avoid the incorrect conclusion.

Existing approaches to IP alias resolution can be classified into two main groups as (1) probe-based approaches and (2) inference-based approaches. Given two IP addresses, probe-based approaches send direct probes to these IP addresses and use the returned responses to decide if the two IP addresses are aliases, i.e., if they belong to the same router or not. The current state-of-the-art probe-based alias resolution approach, `ally` [19], uses IP addresses and IP identification field values of the returned response messages to decide on alias relationship between the given IP addresses. `Ally` introduces a quadratic probing overhead, i.e., $O(n^2)$, for testing alias relations among n different IP addresses.

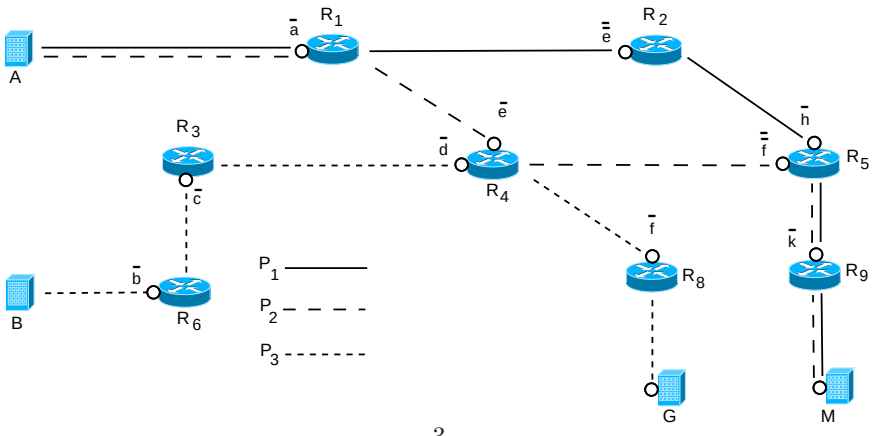
Inference-based approaches leverage certain topological relations that are obtained in the collected topology information to make inferences about possible IP alias relations within the data. As an example, APAR [5] searches for symmetric path segments in `traceroute` returned path traces and aligns them properly to infer IP aliases. For proper alignment, it assumes that most links are point-to-point links and the router



(a) A Network topology section between hosts A, B, G, and M with equally weighted links



(b) Traceroute view of the paths without IP alias resolution. Acquired two shortest paths between hosts A and M are $P_1 = \{A, \overline{a}, \overline{\overline{e}}, \overline{h}, \overline{k}, M\}$ and $P_2 = \{A, \overline{a}, \overline{e}, \overline{\overline{f}}, \overline{k}, M\}$, respectively and a single shortest path between hosts B and G is $P_3 = \{B, \overline{b}, \overline{c}, \overline{d}, \overline{f}, G\}$



(c) Traceroute view of the same paths P_1, P_2, and P_3 with IP alias resolution

Figure 1: **IP Alias Resolution Motivating Example.** An example network segment among hosts A, B, G, and M and traceroute views of the two paths between A and M as well as a single path between B and G with and without IP alias resolution, respectively.

interfaces at both ends of these links are assigned IP addresses from a /30 (or /31) subnet prefix. Compared to probe-based approaches, inference-based approaches do not require additional probing overhead. On the other hand, these approaches significantly depend on the nature and limitations of topology data collection process.

In this paper, we present `palmtree`, a new probe-based IP alias resolution technique with a linear probing complexity. `Palmtree` accepts a list of IP addresses from a target network (or IP address range of the target network) as input and tries to identify IP aliases among the IP addresses of the target network. During this process, `palmtree` leverages the common IP address assignment practices to infer IP aliases (RFC 4632). Compared to existing approaches, `palmtree` proposes an orthogonal approach to IP alias resolution and therefore complements the existing approaches in improving the overall success rate of IP alias resolution process. Compared to `ally`, `palmtree` introduces a linear probing overhead and compared to APAR, it does not require collections of huge volumes of path traces. Yet, empirical results over Internet2 and GEANT backbones as well as four major ISPs indicate high success rates for `palmtree`.

The remainder of the paper is organized as follows. Next section introduces the related work. Section 3 details the `palmtree` algorithm. Section 4 presents the empirical results and evaluations. Finally, Section 5 concludes the paper.

2. Related Work

Several probe and inference based IP alias resolution techniques have been proposed in the literature. In this section, we briefly present some of the most important ones.

`Mercator` [4] is a probe-based IP alias resolver that depends on the similarity in IP addresses of the returned probe responses. If a probe to an IP address I_1 results in a response with a different source IP address I_2, `mercator` sets those two IP addresses as aliases. Additionally, if two probes to two different destination IP addresses I_1 and I_2 results in two response messages with the same source IP address I_3 all IP addresses (I_1, I_2, and I_3) are considered to be aliases. `Mercator` depends on routers replying with a different IP address to probes directly sent to one of its interfaces. However, in practice, routers usually use the probed IP address while responding back to a probe directed to one of its interfaces.

`Ally` is another probe-based alias resolution tool that extends `mercator` by utilizing the IP Identification field values of the returning response packets to decide on aliases. `Ally` exploits the observation that when two IP addresses on the same router are probed back to back, the IP identifier fields of the response messages should be sequential and close to each other. This technique depends on the assumption that routers use a single monotonically increasing IP identification counter while generating responses. Yet, it is known that not all routers use a single IP identification counter across different network interfaces [5]. It is also known that some routers generate IP identification numbers randomly as to minimize the amount of local information leaked out of the router [3]. Finally, due to its dependence on dynamically changing IP identification values, `ally` requires $O(n^2)$ probing overhead when used for resolving IP aliases among a given set of n IP addresses.

`Radargun` [3] employs velocity modeling scheme to reduce `ally`'s quadratic probing complexity. It requires querying of each IP address for a certain period of time and

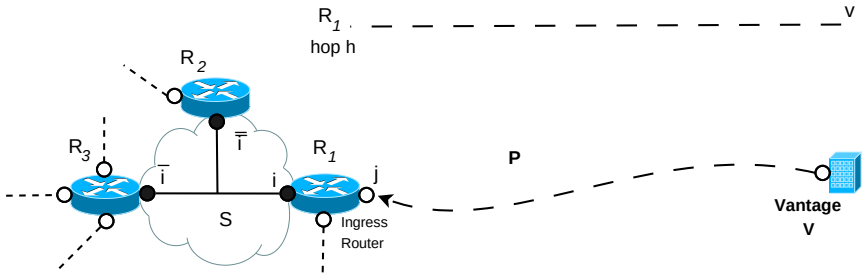


Figure 2: **Palmtree Working Example.** Palmtree first sends a direct distance query packet to estimate the distance of i^{ip}. Then, it sends an indirect alias query packet to $\overline{i}^{ip}$ with $TTL = i^d$ to acquire an alias of i

modeling the IP identification value changes as linear functions. This technique increases the probing budget. In addition, the dynamic nature of the traffic generated by routers at the time of monitoring can lead to potential inaccuracies.

`APAR` [5] is an inference-based alias resolver. It is effective in resolving aliases among IP addresses collected via traceroute during a topology mapping study. `APAR` studies path traces to detect symmetric path segments between two different path traces and uses this symmetry to infer IP aliases. `APAR` leverages a commonly used IP address assignment schemes in the Internet to infer IP aliases without requiring active probes.

`Discarte` [18] is another inference-based alias resolver. It sets the IP route record option in `traceroute` probe packets to determine an IP address as well as one of its aliases. After collecting the topology information, the tool involves a very heavy computational procedure to resolve IP aliases [18]. Success of this scheme significantly depends on limitedly supported IP route record option.

`Palmtree`, presented in this paper, is a lightweight, probe-based IP alias resolver that complements the existing schemes. It uses the common IP address assignment scheme in the Internet to carefully design its probe packets to reveal out IP alias relationship in a given target network. Compared to `ally`, `palmtree` incurs a linear, i.e., $O(n)$, probing overhead and compared to `APAR`, it does not require pre-collection of large number of traceroute-based path traces across a given target network.

3. Palmtree

This section first presents the notations used in the rest of the paper and then introduces a working example of `palmtree` using Figure 2.

We use the following notations: subnets/routers are shown with upper-case letters and are defined as mathematical sets with elements being the interfaces that they accommodate; an interface i is shown by a lower-case letter and has two attributes: an associated IP address shown by i^{ip} and a distance with respect to a certain vantage point v in terms of hop count shown by i^d_v. Whenever the vantage point v is obvious in a context, we drop v and use i^d.

In order to find an alias j of a given IP address i, `palmtree` applies three steps: *(1)* check whether i is assigned to an interface (i.e., i is alive) or not; *(2)* if i is alive,

determine the distance (hop count) to the interface from the vantage point; and *(3)* send a special query packet that will force the router hosting i to report back the IP address of one of its interfaces based on its response configuration. In the following, we present a brief overview and the rest of the section will present the approach in more detail.

Steps (1) and (2) above can be combined and achieved with a single UDP probe packet. A responsive router sends back an ICMP_PORT_UNREACHABLE message as a response to a UDP probe packet. Regardless of the source address field of the response message, the originator of the probe packet (vantage point) infers that the probed IP address is alive. Additionally, IP header plus the first eight bytes of the original probe packet is piggy-backed with the response message as specified by ICMP protocol. Using TTL field of the piggy-backed message one can easily estimate the hop distance from the vantage point to the probed interface. Determining the aliveness status and hop distance of an IP address is referred as *distance querying* in the rest of the paper and the packet sent is called distance query packet.

Step (3) of the algorithm is forcing the router hosting the investigated IP address i to send back a packet whose source address field is set to one of its other IP addresses. To carry out this task, we prepare a packet destined to an IP address being accommodated by the same subnet which accommodates i in a way that the packet would expire at the router hosting i and let the router report an IP address by returning an ICMP_TTL_EXCEEDED message. Particularly, we send an ICMP_ECHO_REQEST packet destined to /31 or /30 mate of i with TTL value set to the hop distance i^d which was obtained by distance querying. Obtaining an alias of an IP address by sending an ICMP message will be referred as *alias querying* in the rest of the paper and the packet sent is called alias query packet.

We now explain the procedure on an example given in Figure 2. In the figure subnet S accommodates $\{i, \overline{i}, \overline{\overline{i}}\}$ and $i, j \in R_1$. Let us assume that R_1 is configured to report incoming interface's IP address as the source address field of generated ICMP_TTL_EXCEEDED messages (please see definition-iii for a detailed discussion on router response configurations) and let i^{ip} and $\overline{i}^{ip}$ be /31 or /30 mates of each other. Given that `palmtree` is fed with i^{ip}, first it sends a distance query packet destined to i^{ip} to locate the hop distance towards the interface i which is h in the figure. After acquiring hop distance, it sends an alias query packet destined to $\overline{i}^{ip}$ with $TTL = i^d = h$. Under a fixed ingress router to the subnet from vantage point v, the alias query packet will expire at R_1 and, as a response, R_1 will generate an ICMP packet with source address set to j^{ip}. Hence, `palmtree` infers that i^{ip} and j^{ip} form an alias pair.

In the following, we first present necessary formal definitions and observations. Next, we give an informal proof of our method using the given definitions and observations and demonstrate the algorithm. Finally we clarify issues such as frontier interface resolution, behavior under path fluctuations, variations in implementation, and probing complexity.

3.1. Definitions

In this section we give a set of definitions which are used in Section 3.3.

(i) Direct Probing is the process of sending a probe packet with large enough TTL value destined to some IP address. It is used to test the existence of an IP address. In general the probe packet is an ICMP ECHO_REQUEST packet or a UDP packet destined to a likely unused port number. The former forces a responsive router to send back an

ICMP ECHO_REPLY and the latter an ICMP PORT_UNREACHABLE packet. As an example using Figure 2, sending a probe packet to IP address $i^{ip} \in R_1$ with a TTL value of at least h causes R_1 to send a proper message back to the originator of the probe.
(ii) Indirect Probing is the process of sending a probe packet with a small TTL value destined to some IP address in order to reveal an IP address of another router presumably located at TTL hops away on the path. The probe packet could be of type ICMP, UDP, or TCP. Whenever the TTL reaches zero a responsive router would notify the originator of the probe message with an ICMP_TTL_EXCEEDED packet. The source address of this packet would be one of the IP addresses of the router based on its response configuration as explained next. As an example using Figure 2, sending a probe packet to IP address $\overline{i}^{ip} \in R_3$ with a TTL value of h causes the message to expire at R_1 and force it to send back an ICMP_TTL_EXCEEDED message to the originator of the probe.
(iii) Router Response Configuration implies that a router is configured to remain reticent or reveal a certain interface's IP address in its response to a direct or indirect probe (query) packet. To the best of our knowledge routers on Internet are configured with five different policies: *nil interface routers* are configured not to respond to any probe packet; *probed interface routers* respond with the address of the probed interface; *incoming interface routers* respond with the address of the interface through which the probe packet has entered into the router; *shortest-path interface routers* respond with the address of the interface that has the shortest path from the router back to the probe originator; and *default interface routers* respond with a pre-designated default interface regardless of the interface being probed. Usually responsive routers are configured to behave as *probed interface routers* for direct probes and any other configuration for indirect probes. Observe that a router cannot be configured as *probed interface router* for indirect queries. Additionally, routers may be configured with multiple response configurations with respect to protocol type of a probe packet.
(iv) Alias Relation; Let $\mathcal{A}$ be the alias relation then ${}_i\mathcal{A}_j$ implies interfaces i and j are hosted by the same router. Note that $\mathcal{A}$ is a symmetric and transitive relation.

3.2. Observations

This section presents a set of observations on Internet which are leveraged in our algorithm.
(i) Hierarchical Addressing details the common IP assignment practices and refers to the Classless Inter-Domain Routing (CIDR) on Internet (RFC 4632). Given any subnetwork S on the Internet, IP addresses assigned to the interfaces on S should share a common p bits prefix. Such a subnet S is said to have $/p$ prefix (subnet mask) and is shown as S^p.

Most of the subnets in the non-edge Internet are point-to-point links having subnet prefix of /30 or /31[7]. Any two IP addresses that have 31 or 30 bits common prefix are called *mate-31* or *mate-30* of each other.
(ii) Fixed Ingress Router implies that as long as there is no path fluctuations caused by routing updates, load balancing, or equal cost multi-path routing, there is a single path from a vantage point to any interface on a given subnet S. As a result two immediately successive probe packets released from a vantage point and destined to two different interfaces on a subnet S are expected to reach the subnet through the same router. This fixed router is called *ingress router*. In Figure 2, R_1 is the ingress router of the subnet S with respect to vantage v (see Section 3.4 for the effects of path fluctuations).

Fixed ingress router is also resistant to intermediate path fluctuations as long as the fluctuated routes converge at or before the ingress router.

(iii) Unit Subnet Diameter implies that for $i, j \in S \Rightarrow |i_v^d - j_v^d| \leq 1$. That is, two interfaces on the same subnet are located at most one hop distance apart with respect to a vantage point. In Figure 2, interfaces $\{i, \overline{i}, \overline{\overline{i}}\} \in S$ are at most one hop apart from each other with respect to v.

(iv) Mate-31 Adjacency implies that given i^{ip} and j^{ip} are alive and mate-31 of each other, then $i \in S \Rightarrow j \in S$.

3.3. Palmtree Algorithm

In this section we first give the justification of `palmtree` algorithm by grounding our arguments on the definitions and observations discussed above and then demonstrate the algorithm.

Definitions (ii) and (iii) state that when the TTL field of an indirect probe packet reaches zero at a responsive router, the router sends back an ICMP_TTL_EXCEEDED message to inform the originator about the drop of its packet. Let i be the interface with the investigated IP address determined to be alive on some subnet S, hosted by some router R, and located at h hops distant from the vantage. Let R be the ingress router of subnet S. That is, i^{ip} is alive, $i^d = h$, $i \in S$, and $i \in R$.

Observations (i) and (iv) say that if $\overline{i}$ is another alive interface and $\overline{i}^{ip}$ is mate-31 of i^{ip}, then $\overline{i} \in S$. Observation (ii) states that since R is the ingress router of subnet S then any probe packet to interface $\overline{i}$ must pass through R. Observation (iii) asserts that if i is hosted by ingress router R and $i^d = h$ then $\overline{i}^d = h + 1$ with respect to the same vantage point. Definition (ii) indicates that a probe packet destined to $\overline{i}^{ip}$ with $TTL = h$ will not make it to reach its final destination (remember that $\overline{i}^d = h+1$) and the responsive router at which TTL reaches to zero respond with an ICMP_TTL_EXCEEDED message. One can see that this probe packet destined to $\overline{i}$ reaches to zero at router R which is hosting i (the investigated interface). Definition (iii) tells that responsive router R reports one of its interfaces, say j, as source IP address field of the ICMP_TTL_EXCEEDED response message. *As a result, interfaces i and j are located on the same router R and ${}_i\mathcal{A}_j$.*

Note that the process depicted above is based on indirect probing of mate-31 of an IP address i with $TTL = h$ where $i^d = h$. The same arguments are valid for mate-30 of i only if indirect probing mate-31 does not yield a response message or yields an ICMP_{HOST/DESTINATION}_UNREACHABLE response.

Compared to `mercator`, `palmtree` sends an indirect probe packet destined to the mate-31 of an IP address i crafted to expire on the router hosting i and force it to report back an alias of i. For the same scenario, `mercator` sends a direct probe packet destined to i and expects the router hosting i to reply with an alias of i. As discussed in Section 3.2 (Router Response Configurations), routers tend to reply back with the probed IP address for direct probes. This is essentially a limitation for `mercator`. `Palmtree` overcomes this limitation by forcing the router to respond with a potentially different, i.e., alias, IP address. It achieves this by carefully constructing and sending an indirect probe packet which expires at the router under investigation. Evaluations section demonstrates that `palmtree` finds more IP alias pairs than `mercator`.

To illustrate the idea using Figure 3 with respect to router R_0 with interfaces $\{a, b, c, d\}$, subnet S_2 with interfaces $\{b, \bar{b}\}$, and $b^d = h$: let b^{ip} be the investigated IP address and

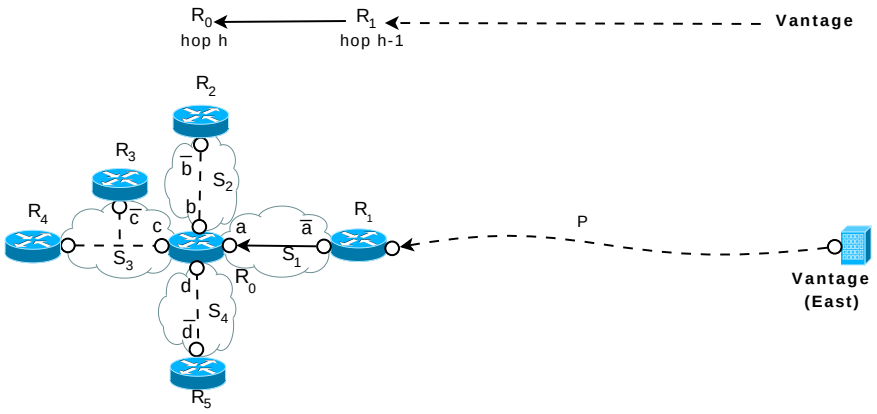


Figure 3: **Palmtree.** For each interface $i \in R_0$ where R_0 is also the ingress router of the subnet on which i is accomodated, `palmtree` acquires ${}_i\mathcal{A}_j$ alias pair where $j \in R_0$

b^{ip} and $\bar{b}^{ip}$ be mate-31 of each other. `Palmtree` algorithm takes the IP address range of a target network or the list of in-use IP addresses of a target network as input and processes each and every IP address one by one. In the scenario depicted in Figure 3 we assumed that the current IP address being processed is b^{ip}. Additionally, without loss of generality let R_0 be a responsive router configured to response through some default interface and let d be the default interface. That is, R_0 *reports* d^{ip} *for indirect probes conducted at the vantage*. Clearly, $\bar{b}^d = h+1$ and any probe packet destined to $\bar{b}^{ip}$ passes through R_0. After determining the existence of b and its distance from the vantage, if one sends a probe packet from the same vantage to mate-31 of b^{ip} ($\bar{b}^{ip}$) with $TTL = b^d = h$ then TTL field of probe packet reaches zero at R_0. R_0 then reports back the situation by sending an ICMP_TTL_EXCEEDED message with source IP address field set according to its response configuration (d^{ip}). Therefore, we infer that ${}_b\mathcal{A}_d$.

Now let us extend the basic principle of finding a single alias pair introduced above to discovering multiple/all IP aliases belonging to R_0. In Figure 3, we apply `palmtree` algorithm to interfaces of $R_0 = \{a, b, c, d\}$. Note that given IP address range of a target network as input, `palmtree` eventually processes all IP addresses hosted by a particular router on the network, however, processing IP addresses of the router is not necessarily sequential. Let R_0 be the ingress router for S_2, S_3, and S_4 in the picture and any packet destined to one of those subnets from the vantage point takes some path P and then reaches R_0 through S_1. For each $i \in R_0$ we also label the interfaces with $\bar{i}$ where i^{ip} and $\bar{i}^{ip}$ are mate-31 or mate-30 of each other. Notice that the subnets are not necessarily point-to-point links, e.g., see S_3. Additionally, $b^d = c^d = d^d = h$ and $\bar{b}^d = \bar{c}^d = \bar{d}^d = h+1$. Applying the basic principle to each and every IP address of R_0 reveals ${}_b\mathcal{A}_d$, ${}_c\mathcal{A}_d$, and ${}_d\mathcal{A}_d$. By utilizing transitivity property of relation $\mathcal{A}$ we can conclude that $\{b, c, d\}$ are aliases of each other.

Apparently, we missed interface a because it shows a totally different behaviour. After determining the distance to a^{ip} from the vantage point, if we send an ICMP_ECHO_REQUEST probe packet to $\bar{a}^{ip}$ with $TTL = h$ (alias query packet), the packet would reach its desti-

nation on R_1 because $\bar{a}^d = h-1$. As a result, R_1 would return an ICMP_ECHO_REPLY. *Regarding the context, a router demonstrates this behaviour for a subnet that it has an interface on but is not the ingress router for it with respect to a certain vantage point.* Such interfaces of a router are called *frontier interfaces* and a router hosts at least one frontier interface with respect to a vantage point. To fix this situation we just track those IP addresses that returned an ICMP_ECHO_REPLY (frontier interfaces) and process them in a second round from a different vantage point particularly positioned in the opposite direction of the first vantage point. In Figure 3 if we repeat the basic `palmtree` process just for IP address a^{ip} from a vantage point located at west, we expect R_0 to report d^{ip} to conclude ${}_a\mathcal{A}_d$. Combining the results of both rounds, we obtain the full alias relationship among $\{a, b, c, d\}$. Note that for some IP addresses the process may take a few rounds conducted at different vantage points.

The success of resolving frontier interfaces depends on the selection of the new vantage point in the second round. Particularly, we need to select a vantage point whose probe packets reaches to the frontier interface i, $i \in S$, $i \in R$ where R is the ingress router for subnet S. The best vantage point for our purposes would be one within the target AS but located oppositely from the previous vantage point. In case there is no available vantage point in the target AS, the best strategy would be selecting a vantage whose packets enter into the target AS through a different border router than the previous vantage point. This new vantage point should be selected from a provider, a customer or a peer AS or an AS which has paths through those ASes towards the target AS.

Algorithm 1 presents the `palmtree` pseudocode. The algorithm takes the IP address range T of a network N as input. Let $DISTANCE(I)$ be a function taking an IP address I as argument and returning the distance (hop count) to I from the vantage point. $DISTANCE(I)$ returns 0 in case the router hosting I is not responsive. $MATE31(I)$ and $MATE30(I)$ are functions taking an IP address I and returning its mate 31 and mate 30 addresses respectively. Let $PROBE(I, d)$ be a function taking an IP address I and a TTL value d, sending a probe packet to this address with $TTL = d$ and returning the resulting response.

Algorithm 1 starts with a for loop and tests existence of each and every IP address within the IP address range T. One can also use the IP addresses that are already known to be alive in order to reduce the search space.

Line 2 explicitly tests the existence of IP address I and calculates the distance from the vantage to I. In lines 4-5, an ICMP probe packet is sent to mate-31 of the subject IP address I. In lines 7-12, the algorithm decides on whether the source address field of the obtained response packet is an alias of the subject IP address or the subject IP address is a frontier IP address and needs to be tested again from some other vantage point. Lines 14-21 are nothing but the repetition of the process explained above with mate-30 of the subject IP address in case probing mate-31 did not return any response. Lines 26-28 show the process of applying the alias transitivity rule in order to extend the alias relationship discovered between pairs in the first part of the algorithm.

`Palmtree` algorithm has the pause-play capability in the sense that for a given set of target IP addresses, one can pause the algorithm before processing the next target and play it later. That is, except within the period of processing single target IP address, `palmtree` does not depend on the current state of routing and traffic on the network.

Algorithm 1 PALMTREE

```
Input: T /* IP address range or in-use IP address list of the subject network */
 1 for each IP address I ∈ T do
 2    d ← DISTANCE(I)
 3    if d > 0 then
 4       Ī ← MATE31(I)
 5       RPLY ← PROBE(Ī, d)
 6       if RPLY ≠ NULL or RPLY ≠ ICMP_HOST/DESTINATION_UNREACHABLE then
 7          if RPLY is ICMP TTL_EXCEEDED then
 8             D ← source IP address of RPLY
 9             IA_D
10          else if RPLY is ICMP ECHO_REPLY then
11             Use a different vantage point for frontier IP address I
12          end if
13       else
14          Ī̄ ← MATE30(I) or (MATE30(MATE31(I)) if I is a border address)
15          RPLY ← PROBE(Ī̄, d)
16          if RPLY ≠ NULL and RPLY is ICMP TTL_EXCEEDED then
17             D ← source IP address of RPLY
18             IA_D
19          else if RPLY ≠ NULL and RPLY is ICMP ECHO_REPLY then
20             Use a different vantage point for frontier IP address I
21          end if
22       end if
23    end if
24 end for
25
26 for all IP alias pairs found do
27    Apply transitivity rule
28 end for
```

3.4. Behaviour under Path Fluctuations

Path fluctuations occurring in the underlying network affects the path taken by two packets towards the same destination or different destinations in the same subnet. `Palmtree` algorithm sends two successive packets destined to two different IP addresses on the same subnet. The first packet, distance query, is for obtaining the distance to an IP address from a vantage point and the second packet, alias query, is for revealing an alias of the IP address. In this subsection we discuss the cases where distance query and alias query (i) take different paths and (ii) enter into the subnet through different ingress routers due to a precedent load balancing router.

In scenario (i), a load balancing router appearing on the path towards the destination subnet may send the distance and alias query packets over different paths but they enter to the subnet through the same ingress router. That is, both paths converge to the same ingress router. This type of load balancing has no harm on the success of `palmtree` as long as the hop counts of both paths are the same (symmetric diamonds). On the other hand, if hop counts of the paths vary (asymmetric diamonds), `palmtree` might introduce false positives. Recent work on load balancing demonstrates that the number of asymmetric load balancers in the Internet is very small [2].

As for scenario (ii), Figure 4 shows two responsive ingress routers R_0 and R_1 to subnet S located at h hops distant from the vantage point. Let both routers be incoming interface routers and the target IP address to be investigated be b^{ip}. As shown in the figure, the immediately successive distance and alias query packets take different paths. Since the alias query expires on router R_1, R_1 reports back the incoming interface's IP address e^{ip} to the originator of the query. As a result the originator misconcludes that

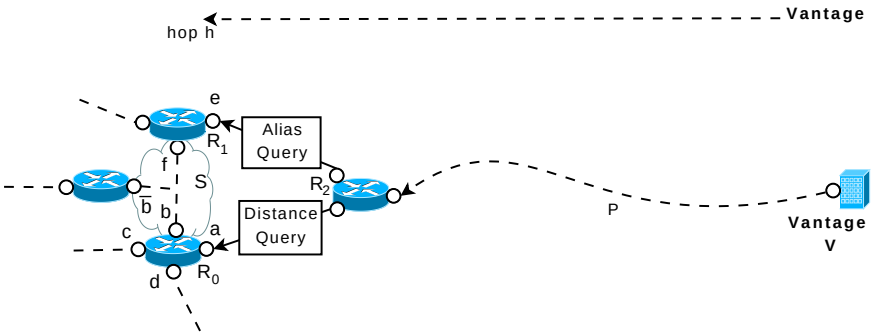


Figure 4: **Palmtree with Unfixed Ingress Router.** Distance and Alias query packets entering into the subnet through multiple ingress routers results in false positive alias pairs.

${}_b\mathcal{A}_e$.

Note that, despite a single load balancer may potentially corrupt all `traceroute` paths passing through it, its effect is not the same in `palmtree` because *palmtree is based on the stable ingress router concept not stable path trace concept.*

Path fluctuations problem can be addressed by increasing confidence level on an obtained alias pair in both scenarios. To increase our confidence level, after obtaining the distance to a target IP address, we send two or more alias query packets and confirm that the source address field of all response messages are the same. In case we get a different source address we can either use another vantage point or resort other IP alias resolution techniques like `ally` for that particular target IP address. An execution of `palmtree` algorithm with confidence level k is simply executing lines 4-22 of Algorithm 1 k times.

3.5. Variations in Implementation

First of all, [9] shows that most responsive routers are configured to be responsive to direct ICMP probes while remain silent to UDP or TCP probes. To get the fullest benefit from routers one can use ICMP probe packets by going forwards and backwards from a certain TTL value to exactly locate the hop distance to an IP address. We implemented both UDP and ICMP based distance discovery functions into `palmtree`.

Secondly, direct UDP distance probes may result in an ICMP_PORT_UNREACHABLE response message with a source IP address different from the probed one. Consequently, the probed IP address and reported IP address are alias. This method is called *source based alias resolution* and implemented in `mercator`. Despite being limitedly implemented on routers, we took advantage of this method because it does not incur any additional probing cost to us.

Finally, `palmtree` could be embedded into `traceroute` as an attempt to reveal an alias at each hop on the path trace.

3.6. Probing Complexity

`Palmtree` algorithm requires distance and alias querying. Alias querying is always a single probe, on the other hand, cost of distance querying varies as follows: UDP distance query requires a single probe, ICMP distance query requires a constant number of probes between 1 and radius of Internet.

There are two additional components of `palmtree` contributing to the total probing overhead. First one is the frontier interface resolution which requires running `palmtree` at different vantage point(s) for particular IP addresses. Second one is the confidence level fortification which requires repeating alias probing process for a constant number of times.

For a given target AS with n utilized IP addresses, let us assume that `palmtree` detects aliases for $1/m$ of the whole target set and leaves $1 - 1/m$ portion for the next round as frontier interfaces where m is a constant. Also let the confidence level be constant k and average distance querying overhead be q. Then probing overhead is:

$$\sum_{r=0}^{\infty}(k+q)n(1-\frac{1}{m})^r = (k+q)mn \Rightarrow O(n). \quad (1)$$

4. Evaluations

In this section, we present our evaluations of the `palmtree` algorithm. We use `palmtree` on Internet2 and GEANT, to measure its accuracy and completeness. We also run `palmtree` on four major commercial ISP topologies including Sprint, AboveNet, Level3, and NTT America to measure its accuracy and compare the performance of `palmtree` to that of `mercator` and `ally` on commercial ISP topologies. Finally, we measure the accuracy rate of `palmtree` over a set of multicast enabled routers scattered around the world which is collected by `mrinfo` tool[11].

Internet2 and GEANT topology maps are publicly available on the Internet giving us the ground truth to make accuracy and completeness evaluations. In this set of experiments we used three different vantage points to resolve frontier interfaces (see Section 3.3 for details on frontier interfaces). Using PlanetLab nodes as vantage points worked well while discovering the Internet2 and GEANT backbone. However, our experiments on commercial ISPs revealed that most of the query packets originated from different PlanetLab vantage points follow a distinct upstream path and then follow a common downstream path towards the target ISP and enter into the target ISP through the same border router. Hence, incautiously selected vantage points might not provide additional gain in resolving frontier interfaces. As previously indicated, the best strategy to resolve frontier interfaces is to find available vantage points at provider, customer, or peer ASes of the target AS. Therefore, we used single vantage points in the last two experiments which is quite consistent considering that the aim of the experiments is to demonstrate the accuracy rate of `palmtree`. Besides, the number of acquired alias pairs is significantly larger than the other methods even with a single vantage point. Nevertheless, we believe that deploying `palmtree` on Skitter/Ark[12] vantage points or embedding it into DIMES[17], a distributed Internet topology collector, would significantly increase the number of aliases found by `palmtree`.

4.1. Palmtree over Internet2

In this experiment, we aimed to measure the success rate and completeness of `palmtree` algorithm over Internet2, a network whose backbone topology is known to us. The experiment consists of three rounds where each round is executed at a different PlanetLab site. Since Internet2 nodes are configured not to respond UDP probes, we used ICMP distance probing in `palmtree`. Comparing results of `palmtree` with that of `ally` and `mercator` is not applicable because their implementations are based on UDP probing.

Results show that 85% of IP addresses immediately resolved at the first round; 95% of IP addresses correctly distributed to their associated routers at the end of three rounds; and 5 out of 9 routers are resolved completely.

As our experimental setup we first obtained the IP addresses of each router on Internet2 backbone[1]. In order to have a list of IP addresses that are observed to be in use we cleared out those ones that report unreachable (ICMP type 3) to direct probes or do not report anything at all. We designated University of Texas at Dallas (south), University of Massachusetts (east), and University of Washington (west) PlanetLab sites as our first, second, and third round vantage points. Note that the whole IP address list is only given to the first round vantage point as input. The second round vantage point is fed with only those IP addresses determined to be frontier in the first round and the third vantage point is fed with those IP addresses still remained as frontier in the second round. We ran `palmtree` with confidence level 1 and classified the results into four categories per round; successfully resolved IP addresses, IP addresses that are mistakenly assigned to a router (false positives), IP addresses that are not assigned to its router (false negatives), and frontier IP addresses.

Table 1 details the results obtained over Internet2. The table presents the results at the end of each round and gives the total figures. The first column (Router) lists the router names on the Internet2 backbone, the second column (RS) show the number of responsive IP addresses for each router. Per each round there are five columns: first column (IN) is the number of Input IP addresses processed at the particular round. Second column (SC) is distribution of the successfully resolved IP aliases per router. Third (FP) and fourth (FN) columns reflect the number of false positive and false negative IP addresses per router. The last column (FR) presents number of frontier IP addresses per column. The very last column group in the second table summarizes the total.

In the table, the high majority of the IP aliases are resolved at the first round. This behavior stems from the fact that the published topology of Internet2 is a backbone topology. In order to empirically demonstrate the independence between the first round vantage point and number of detected IP aliases, we repeated the first round of the same experiment from University of Massachusetts instead of University of Texas at Dallas; the results show that 165 of 212 target IP addresses are resolved at this single round. In general, although the number of input target addresses drop very fast at the first few rounds, it turns out to be harder to locate an appropriate vantage point for those small number of target IP addresses remained to be frontier.

[1] http://www.internet2.edu/observatory/archive/data-collections.html

Table 1: `Palmtree` Results over Internet2 at UTDallas, UMass, and UW rounds as well as totals

		Round 1 (UTDallas)					Round 2 (UMass)					Round 3 (UW)					**Total**				
Router	RS	IN	SC	FP	FN	FR	IN	SC	FP	FN	FR	IN	SC	FP	FN	FR	RS	SC	FP	FN	FR
seat	10	10	5	0	1	4	4	1	0	0	3	3	2	0	0	1	10	8	0	1	1
newy32aoa	34	34	29	0	0	5	5	3	0	0	2	2	0	0	0	2	34	32	0	0	2
wash	32	32	27	0	0	5	5	4	0	0	1	1	0	0	0	1	32	31	0	0	1
chic	49	49	42	0	0	7	7	1	0	0	6	6	1	0	0	5	49	44	0	0	5
atla	18	18	17	0	0	1	1	1	0	0	0	0	0	0	0	0	18	18	0	0	0
kans	17	17	15	0	0	2	2	2	0	0	0	0	0	0	0	0	17	17	0	0	0
salt	19	19	14	0	1	4	4	2	0	0	2	2	2	0	0	0	19	18	0	1	0
losa	18	18	17	0	0	1	1	0	0	0	1	1	1	0	0	0	18	18	0	0	0
hous	15	15	14	0	0	1	1	1	0	0	0	0	0	0	0	0	15	15	0	0	0
Total	212	212	180	0	2	30	30	15	0	0	15	15	6	0	0	9	212	201	0	2	9

This execution of `palmtree` did not produce any false positives. False positives are produced in the case of a path fluctuation between a distance query packet and the associated alias query packet as explained in Section 3.4.

We further analysed the reason behind the two false negative aliases appearing on "seat" and "salt" routers. It appears that although distance query packets of those two IP addresses returned a successful distance value, their alias query packets destined to mate-31 and mate-30 IP addresses did not yield any response message. Probably, IP address range of the subnets on which those two IP addresses reside were sparsely utilized and mate-31 and mate-30 IP addresses were not in use or the ingress router did not reply due to heavy traffic load at that moment.

4.2. Palmtree over GEANT

In this experiment, we ran `palmtree` over GEANT network in order to support the findings of the first experiment over Internet2 and show that `palmtree` performance does not depend on any particular target network configuration. This experiment also has three rounds executed at three PlanetLab vantage points.

`Palmtree` correctly assigned 95% of the IP addresses to their corresponding routers. 88% of these correctly assigned IP addresses were resolved in the first round. In total, out of 18 GEANT routers, 8 of them were resolved completely, 8 of them were resolved except one IP address, and 2 of them were resolved except two IP addresses. Additionally, out of 318 IP addresses, 304 of them successfully assigned to their routers while 10 of them remained to be frontier and two of them left as false negative at the end of three rounds.

As our setup, we acquired publicly available list of routers and their associated IP addresses of GEANT[2]. We designated PlanetLab sites VTT Technical Research Centre of Finland (VTT), Athens University of Economics and Business (AUEB), and The New University of Lisbon (UNL) as our first second and third vantage points, respectively. We again ran `palmtree` with confidence level 1 and classified the results into four categories per round; successfully resolved IP addresses, IP addresses that are mistakenly assigned to a router (false positives), IP addresses that are not assigned to its router (false negatives), and frontier IP addresses.

Table 2 show the detailed results obtained over Internet2. The table presents the results at the end of each round and gives the total figures. The reading of the tables is the same as the tables of the previous Internet2 experiment so we do not repeat it here again. However, we observed that false negatives (FN) are due to these IP addresses do not reply `palmtree` distance query packets and we fed them as input to the next round. Results show that some of these unresponsive IP addresses responded in the following rounds, we believe that either distance query packets got lost or the routers applied rate limiting.

[2]http://stats.geant2.net/lg/

Table 2: `Palmtree` Results over GEANT at VTT, AUEB, and UNL rounds as well as totals

		Round 1 (VTT)					Round 2 (AUEB)					Round 3 (UNL)					**Total**				
Router	RS	IN	SC	FP	FN	FR	IN	SC	FP	FN	FR	IN	SC	FP	FN	FR	RS	SC	FP	FN	FR
rt1.ams.nl.geant2.net	23	23	17	0	0	6	6	6	0	0	0	0	0	0	0	0	23	23	0	0	0
rt1.ath2.gr.geant2.net	11	11	10	0	0	1	1	1	0	0	0	0	0	0	0	0	11	11	0	0	0
rt1.buc.ro.geant2.net	10	10	9	0	0	1	1	0	0	0	1	1	0	0	0	1	10	9	0	0	1
rt1.bud.hu.geant2.net	19	19	16	0	0	3	3	2	0	0	1	1	0	0	0	1	19	18	0	0	1
rt1.cop.dk.geant2.net	14	14	12	0	0	2	2	2	0	0	0	0	0	0	0	0	14	14	0	0	0
rt1.fra.de.geant2.net	30	30	26	0	1	3	4	4	0	0	0	0	0	0	0	0	30	30	0	0	0
rt1.gen.ch.geant2.net	18	18	15	0	1	2	3	0	0	1	2	3	2	0	1	0	18	17	0	1	0
rt1.kau.lt.geant2.net	10	10	7	0	1	2	3	1	0	1	1	2	0	0	1	1	10	8	0	1	1
rt1.lon.uk.geant2.net	34	34	32	0	0	2	2	1	0	0	1	1	1	0	0	0	34	34	0	0	0
rt1.mad.es.geant2.net	15	15	10	0	1	4	5	2	0	0	3	3	3	0	0	0	15	15	0	0	0
rt1.mil.it.geant2.net	25	25	20	0	1	4	5	3	0	1	1	2	1	0	0	1	25	24	0	0	1
rt1.par.fr.geant2.net	23	23	21	0	0	2	2	1	0	0	1	1	0	0	0	1	23	22	0	0	1
rt1.poz.pl.geant2.net	17	17	15	0	0	2	2	0	0	0	2	2	0	0	0	2	17	15	0	0	2
rt1.pra.cz.geant2.net	11	11	10	0	0	1	1	1	0	0	0	0	0	0	0	0	11	11	0	0	0
rt1.rig.lv.geant2.net	10	10	9	0	0	1	1	1	0	0	0	0	0	0	0	0	10	10	0	0	0
rt1.sof.bg.geant2.net	15	15	12	0	0	3	3	2	0	0	1	1	0	0	0	1	15	14	0	0	1
rt1.tal.ee.geant2.net	10	10	9	0	0	1	1	0	0	0	1	1	0	0	0	1	10	9	0	0	1
rt1.vie.at.geant2.net	23	23	19	0	0	4	4	3	0	0	1	1	0	0	0	1	23	22	0	0	1
Total	318	318	269	0	5	44	49	30	0	3	16	19	7	0	2	10	318	304	0	2	10

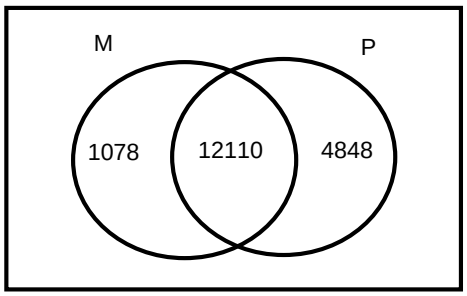


Figure 5: **Mercator vs PalmTree.** Validation of PalmTree by Mercator.

4.3. Palmtree Accuracy over Commercial ISPs

This experiment is used to justify success rate of `palmtree` over commercial ISPs. We prepared a target IP address set T consisting of 92354 IP addresses belonging to four different commercial ISPs (Sprint, AboveNet, Level3, and NTT America). Given that we do not have access to the topology information of these ISPs, we used IP aliases returned by `mercator` as ground truth and compared our results to them. We ran `mercator` over target IP address set T and built a set M representing IP alias pairs found by `mercator`. The number of ground truth IP alias pairs returned by this process is 13188, i.e., $|M| = 13188$.

To capture alias pairs only with alias querying, we disabled source based alias resolution feature of our `palmtree` implementation and ran it from a single vantage poin; fed with the same target set T; and built another set of alias pairs P. The number of alias pairs acquired by `palmtree` is 16958, i.e., $|P| = 16958$. The results in Figure 5 demonstrate that $|P \cap M| = 12110$ and $|P \setminus M| = 4848$. Put in other words, `palmtree` immediately captures 92% of those true IP alias pairs that are found by `mercator`. The set $M \setminus P$ which has 1078 alias pairs are identified to be frontier interfaces by `palmtree` and could be captured by using another vantage point.

Remember that, source based alias resolution feature of `palmtree` has no additional probing cost at all and when enabled $P \cap M$ would always be equal to M. Additionally, when `palmtree` is based on ICMP distance querying it almost doubles the number of discovered IP alias pairs as shown in the next experiment.

Additionally, the distance querying method of `palmtree` in this experiment was UDP. As demonstrated in Section 4.4, repeating the same experiment over the same target set T with ICMP distance querying results in 30381 alias pairs which almost doubles the size of the set P that was built in this experiment.

4.4. Palmtree and Ally

This experiment compares and partially validates `palmtree` with `ally`. First, we ran `palmtree` with ICMP distance querying from a single vantage point located at UT-Dallas over the same target set T with 92354 alive IP addresses used before. After obtaining IP aliases, we test them using `ally` and present the level of agreement and disagreement between the two tools.

Table 3 shows the results collected by `palmtree`. The table demonstrates that among those 92354 target IP addresses; `palmtree` successfully discovered an alias for 30381, marked 37968 as frontier (requires another vantage point to be resolved), and failed to detect an alias for 24005 as the response to either distance query or alias query was missing or was of type unreachable.

Table 3: **Palmtree Results over Commercial ISPs.** Presents distribution of collected data over ISPs after executing `palmtree` at a single vantage point

	Sprint	**NTTAmerica**	**Level3**	**AboveNet**	**Total**
Target	21876	29428	33168	7882	**92354**
Alias	6824	8400	12218	2939	**30381**
Frontier	7983	14779	12142	3064	**37968**
Missing	7069	6249	8808	1879	**24005**

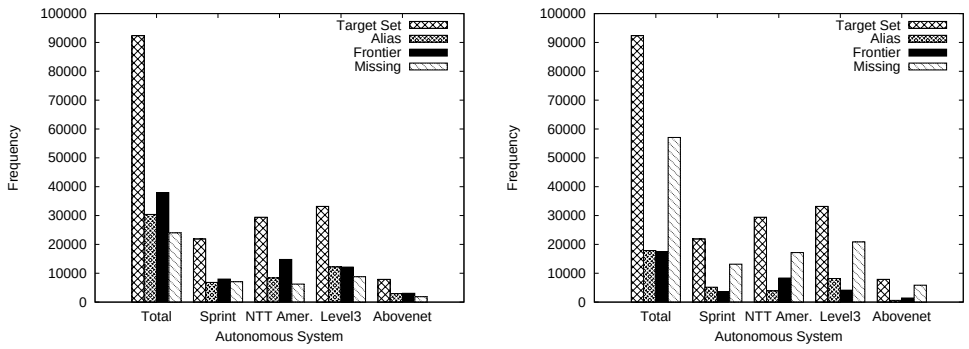
Set P consists of those 30381 IP pairs collected by `palmtree`. As the next step of our experiment, we fed `ally` with each pair $r \in P$ and acquired the decision made by `ally` for all r. Testing a pair of IP addresses by `ally` does not result in a binary yes/no answer. Instead, `ally` reports (i) whether probing the IP addresses resulted in responses with IP identifiers that are in sync or not, (ii) whether probing the IP addresses resulted in two responses having the same source IP address or not, (iii) whether probing one of the addresses results in a response having source address field set to the other IP address or not, and (iv) whether there is an exceptional case such as not obtaining a response from one or both of the IP addresses. To make a reasonable comparison, we categorized `ally` responses for each pair as follows: if case (iv) is true, we marked the pair as **unknown**; if cases (ii) or (iii) are true, we marked the pair as **source-verified**; if case (i) is true, we marked the pair as **IP ID-verified**; and if neither case (i) nor cases (ii) and (iii) are true, we marked the pair as **unverified**.

Table 4: **Ally Results.** Categorization of `ally` results for 30381 IP alias pairs determined by `palmtree`

IP ID-verified	**source-verified**	**unverified**	**unknown**	**Total**
5166	4717	1319	19179	**30381**

The high number of unknown pairs shown in Table 4 results from the fact that `ally` is based on UDP probing whereas `palmtree` is based on ICMP probing and most of the routers are silent to UDP probes while responsive to ICMP probes. Unfortunately, we cannot confirm the collected pairs against real topologies of those ISPs in order to make a judgement about the pairs found to be unverified by `ally`. However, to have an idea that those 1319 alias pairs are not false positives of `palmtree` caused by asymmetric load balancing or unstable ingress router (see Section 3.4), we ran `palmtree` with confidence level 5 over those IP addresses: only 15 pairs resulted in two distinct aliases among five back-to-back alias queries per pair. This suggests that except 15 of these 1319 alias pairs are true positives.

Since this execution of `palmtree` was based on ICMP distance querying, `palmtree` did not return any source based IP alias pairs whereas `ally` verified 4717 of `palmtree` alias pairs by source based alias method and 5166 of pairs by IP identification method.



(a) PalmTree with ICMP Distance Querying (b) PalmTree with UDP Distance Querying

Figure 6: Palmtree Results over Commercial ISPs with ICMP and UDP Distance Querying

Finally, we could not check for false negatives of `palmtree` with respect to `ally` as it would require us to run `ally` for all $\binom{92354}{2}$ IP address pairs which is quite expensive and time consuming.

Figure 6 visually compares the results obtained by ICMP and UDP distance querying over the same target IP address set. *Alias*, *Frontier*, and *Missing* demonstrate the number of acquired alias pairs, frontier interfaces, and IP addresses for which `palmtree` could not obtain a response message to distance and/or alias queries respectively.

Figure 6 confirms previously reported observations that routers are more responsive to ICMP probes rather than to UDP probes[9]. In the figure, the total number of missing IP addresses with UDP is almost three times larger than that of ICMP. Moreover, the total number of frontier interfaces with ICMP seems to be more than the ones with UDP which is not immediately intuitive. The reason behind this is that the frontier interfaces found with ICMP mostly did not respond to UDP distance queries and hence categorized as *Missing* in Figure 6b rather than as *Frontier*.

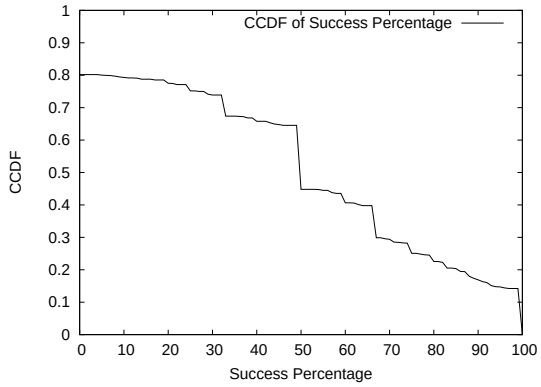
4.5. Palmtree over Mrinfo Data

`Mrinfo` [11] is a multicast diagnostic tool which is used to query a multicast enabled router to learn the list of IP addresses of its multicast qualified interfaces. If a router has enabled multicast service on all of its interfaces, then `mrinfo` essentially reports all the IP addresses belonging to that router. `Mrinfo` has recently been used to collect topology information of multicast enabled routers in the Internet [15]. The limitations of `mrinfo` in terms of the completeness requirement of constructing Internet topology maps are; (1) it works only with multicast enabled routers; (2) it does not always return the complete list of IP addresses of a router; (3) it is prone to information limiting policies and IGMP filtering applied by ISPs. In this part of evaluations, we use `mrinfo` data as ground truth to validate the accuracy of `palmtree` on the public Internet.

In this experiment we used the topology data periodically collected and published at

Pansiot's project web page[3] as our ground truth data. The data consist of 2455 routers scattered around the world with 10484 IP addresses in total. After removing the routers with incomplete information we had 1272 routers with 8268 IP addresses in total. The number of interfaces belonging to routers range from 2 to 72. We designated PlanetLab sites from USA, Finland, Egypt, Argentina, China, and Australia as our vantage points. Note that one factor that is disadvantageous to `palmtree` in this experiment is that the routers obtained by `mrinfo` belong to different ISPs disseminated on the world. However, the standard usage of `palmtree` requires a single ISP as the target network domain and a few vantage points carefully selected to reveal much of the target network. Deriving the complete topology of the Internet involves processing major ISPs one by one.

Using `mrinfo` data as ground truth, `palmtree` successfully assigned 57% of all IP addresses to their associated routers. The rest of the IP addresses were mostly frontier and unresponsive. In terms of routers, `palmtree` failed to resolve 19% of all routers at all and fully resolved 14% of them. Additionally, `palmtree` resolved 65% of all routers with a success rate of at least 50%.



Routers	1272
Interfaces	8268
Successfully Resolved Interfaces	4771
False Positives	136
Frontier Interfaces	1844
False Negatives	1517

Figure 7: **Palmtree Performance over Mrinfo Data.** Raw figures and CDF of success rates of `palmtree`.

We divided percentage range [0-100] into one unit bins and distribute the routers into these bins based on the success rate of `palmtree` for each router. Figure 7 shows the complementary cumulative distribution (ccdf.) of the success rate. In the figure, 19% of routers have zero success rate, i.e., `palmtree` could not resolve these routers at all. On the other hand, 14% of the routers have 100 success rate, i.e., `palmtree` completely resolved those routers. Additionally, the figure shows that 65% of all routers are resolved with at least 50% success rate.

Although ccdf. shown in Figure 7 details the success rate distribution of `palmtree`, summary statistics such as mean and standard deviation are not applicable because we distribute the routers into percentage bins without considering the number of interfaces that a router has. To gain more intuition, we summarize the outcome of the experiment in the table next to the figure. According to the table, the raw fraction of successfully

[3] http://clarinet.u-strasbg.fr/~pansiot/mrinfo/data/

resolved IP addresses by `palmtree` among 8268 total IP addresses is 57%; fraction of false positives is 1.6% and fraction of frontier IP addresses is 22%. Additionally, 18% of IP addresses turned out to be false negatives as these IP addresses did not respond to `palmtree` probes.

In summary, assuming the routers in the Internet share the same features with the routers in this dataset one can argue that given a random, in-use IP address, `palmtree` can find an alias for that IP address with probability 0.57.

5. Conclusions

In this paper, we have presented, `palmtree`, a new probe-based IP alias resolution tool with linear probing overhead. `Palmtree` uses an orthogonal approach to alias resolution and therefore complements the existing IP alias resolution tools in increasing the overall success of alias resolution task during a topology map construction process.

Our evaluations of `palmtree` demonstrated its utility in alias resolution. We conducted five experiments to present performance of `palmtree`. Our first two experiments on Internet2 and GEANT topology both resulted in 95% success rate for `palmtree` when run from multiple vantage points. In our third experiment, we compared `palmtree` to `mercator` to quantify its accuracy when run from a single vantage point. The results showed that `palmtree` was successful in returning about 92% of IP aliases correctly when run from a single vantage point. Besides, compared to `mercator`, `palmtree` has captured 26% and 56% more IP alias pairs with UDP and ICMP distance querying, respectively. Our fourth experiment partially validated the performance of `palmtree` by another probe-based alias resolver `ally`. `Ally` showed conflict on 4%, approved 32%, and could not make a decision on 64% of all IP alias pairs found by found by `palmtree`. The last experiment in which we used a set of routers scattered around the world demonstrated that 65% of all routers in this set are resolved with at least 50% success rate.

Although, we measured completeness of `palmtree` on Internet2 and GEANT our main focus has been accuracy validation in this first study which presents `palmtree`. Nevertheless, we plan to sketch full router level ISP topologies and share our findings with the community as a future work. Finally, an implementation of `palmtree` will soon be made publicly available on our project web site at http://itom.utdallas.edu.

References

[1] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with Paris traceroute. In *Proc. of ACM IMC*, Rio de Janeiro, Brazil, October 2006.

[2] B. Augustin, T. Friedman, and R. Teixeira. Measuring load-balanced paths in the internet. In *Proc. of IMC*, NY, USA, Oct. 2007.

[3] A. Bender, R. Sherwood, and N. Spring. Fixing ally's growing pains with velocity modeling. In *Proc. of ACM IMC*, Greece, Oct. 2008.

[4] R. Govindan and H. Tangmunarunkit. Heuristics for Internet map discovery. In *Proc. of IEEE INFOCOM*, Tel Aviv, Israel, March 2000.

[5] M. Gunes and K. Sarac. Resolving IP aliases in building traceroute-based internet maps. *IEEE/ACM Transactions on Networking*. to appear.

[6] M. Gunes and K. Sarac. Importance of IP alias resolution in sampling internet topologies. In *IEEE Global Internet Symposium*, 2007.

[7] M. Gunes and K. Sarac. Inferring subnets in router-level topology collection studies. In *Proc. of ACM IMC*, CA, USA, Oct. 2007.

[8] M. Gunes and K. Sarac. Resolving anonymous routers in internet topology measurement studies. In *Proc. of IEEE INFOCOM*, Phoenix, AZ, USA, April 2008.
[9] M. Gunes and K. Sarac. Analyzing Router Responsiveness to Active Measurement Probes. In *Proc. of PAM*, Seoul, Korea, April 2009.
[10] V. Jacobson. Traceroute. Lawrence Berkeley Laboratory (LBL), February 1989. Available from ftp://ee.lbl.gov/traceroute.tar.Z.
[11] V. Jacobson. Mrinfo. 1995. Available from http://cvsweb.netbsd.org/bsdweb.cgi/src/usr.sbin/mrinfo.
[12] D. M. K. Claffy, Tracie E. Monk. Internet Tomography. *Nature*, January 1999.
[13] H. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane: An information plane for distributed services. In *Proc. of OSDI*, Seattle, WA, USA, November 2006.
[14] D. McRobb, K. Claffy, and T. Monk. *Skitter: CAIDA's macroscopic Internet topology discovery and tracking tool*, 1999. Available from http://www.caida.org/tools/skitter/.
[15] J.-J. Pansiot, P. Mrindol, B. Donnet, and O. Bonaventure. Extracting intra-domain topology from mrinfo probing. In *Proc. Passive and Active Measurement Conference (PAM)*, April 2010.
[16] Y. Shavitt and E. Shir. DIMES: Distributed Internet measurements and simulations. Project page http://www.netdimes.org.
[17] Y. Shavitt and E. Shir. Dimes: Let the internet measure itself. volume 35, page 74, 2005.
[18] R. Sherwood, A. Bender, and N. Spring. DisCarte: A disjunctive internet cartographer. In *Proc. of ACM SIGCOMM*, WA, USA, Aug. 2008.
[19] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies with Rocketfuel. *IEEE Trans. On Networking*, Feb. 2004.
[20] University of Oregon. Route views project. http://www.routeviews.org.