A Fistful of Pings: Accurate and Lightweight Anycast Enumeration and Geolocation

Danilo Cicalese*, Diana Joumblatt*, Dario Rossi*, and Marc-Olivier Buob†, Jordan Augé†, Timur Friedman†
*Telecom ParisTech, Paris, France – `first.last@telecom-paristech.fr`
†UPMC Sorbonne Universités, Paris, France – `first.last@lip6.fr`

***Abstract*—Use of IP-layer anycast has increased in the last few years: once relegated to DNS root and top-level domain servers, anycast is now commonly used to assist distribution of general purpose content by CDN providers. Yet, the measurement techniques for discovering anycast replicas have been designed around DNS, limiting their usefulness to this particular service.**

This raises the need for protocol agnostic methodologies, that should additionally be as lightweight as possible in order to scale up anycast service discovery. This is precisely the aim of this paper, which proposes a new method for exhaustive and accurate enumeration and city-level geolocation of anycast instances, requiring only a handful of latency measurements from a set of known vantage points. Our method exploits an iterative workflow that enumerates (an optimization problem) and geolocates (a classification problem) anycast replicas.

We thoroughly validate our methodology on available ground truth (several DNS root servers), using multiple measurement infrastructures (PlanetLab, RIPE), obtaining extremely accurate results (even with simple algorithms, that we compare with the global optimum), that we make available to the scientific community. Compared to the state of the art work that appeared in INFOCOM 2013 and IMC 2013, our technique (i) is not bound to a specific protocol, (ii) requires 1000 times fewer vantage points, not only (iii) achieves over 50% recall but also (iv) accurately identifies the city-level geolocation for over 78% of the enumerated servers, with (v) a mean geolocation error of 361 km for all enumerated servers.

I. Introduction

Geolocation of hosts according to their IP addresses is widely performed, both commercially (e.g., MaxMind [1]) and for research purposes. However there is a class of IP addresses on which today's geolocation methods fail: anycast addresses. IP-layer anycast [2] allows a group of *replicas*, or distributed servers that offer the same service, to share a single unicast IP address. Traffic directed to that address gets routed to just one replica, as determined by BGP routing. Geolocation services incorrectly assume that each unicast IP address corresponds to a host or set of hosts at a single location, and are unable to flag instances where this is not the case, not to mention being able to respond to a geolocation query with a set of hosts and locations.

One explanation for this deficit in anycast IP geolocation is that current services are oriented towards geolocation of client machines, not servers. Content distributors might geolocate clients in order to respect contractual constraints that limit them to sending certain content to certain countries; banks might geolocate clients for security purposes and as part of their due diligence concerning knowledge of their customers; advertisers might geolocate clients in order to profile populations and deliver targeted messages. However, more and more services are being delivered through anycast. DNS root and .ORG top level domain services [3], [4] have been anycast pioneers, using the technique to optimize availability and response time. The AS112 project [5], which reduces the load on DNS root servers caused by reverse DNS lookups for private network and link-local addresses, uses anycast. Multicast groups implement anycast mechanisms for rendezvous point discovery [6]. To connect IPv6 networks across an IPv4 infrastructure, 6-to-4 relay routers [7] advertise an anycast IP prefix. Sinkholes [8] use anycast to detect, contain, and analyze worm activity. And now we are seeing commercial services such as content delivery networks (CDNs) increasingly being offered through anycast. For example, the EdgeCast CDN [9], which supports such major services as Pinterest and Twitter, is entirely anycast. With this rise of anycast, there is a concomitant need to understand anycast services. Internet service providers have a large commercial stake in managing over-the-top content flows, for example, and want to know where these flows are coming from and why. Businesses that rely upon services that are delivered via anycast need adequate troubleshooting tools. The locations from which services are provided are of interest to scientists ranging from security researchers to economists and social scientists, and of course network researchers.

Prior work on identifying, enumerating, and geolocating anycast services [10], [11] has focused entirely upon DNS. The techniques used are specific to DNS and are not generalizable to other services, as they rely upon DNS CHAOS [11] or EDNS [12]. The contribution of this paper is that it provides a general technique for determining whether services are being offered via anycast on any unicast address. Further, it shows how to enumerate the replicas that are behind an anycast address and how to geolocate those replicas.

The geolocation methodology described in this paper also differs significantly from prior IP address geolocation work [13]–[17]. Like [13], it is based upon speed-of-light constraints. But the technique that consists in using multilateration to infer the location of a single host is designed for single-host unicast and fails with anycast. We put an original twist on the technique, to adapt it to the situation where there may be multiple hosts. A simple image conveys the difference of our approach: that of discs that cover areas on a map. Single-host unicast multilateration employs discs centered around multiple vantage points, and locates the target host somewhere in the area formed by the intersection of the discs. In our technique, we recognize anycast in cases where such discs do not overlap, and we position each host at a large population center within a disc. Our methodology consists in (i) framing the enumeration task as a maximum independent set optimization problem, that we optimally solve, (ii) framing the geolocation task as a classification problem, where we exploit side information such as population density, and (iii) solving both problems iteratively, with output of the geolocation task fed back to the enumeration task, which provides sizeable benefits.

The remainder of this paper is organized as follows. We first state our objectives in Sec. II, and then explain both our overall workflow as well as the enumeration and geolocation techniques in Sec. III. Validation and calibration of these techniques over a dataset gathered in PlanetLab is the object of Sec. IV. We then run our algorithm on a fully fledged measurement campaign from multiple measurement infrastructures in Sec. V, comparing our results to state of the art techniques. We make these datasets available to the scientific community at [18]. Finally, Sec. VI overviews related work and Sec. VII gathers conclusive remarks and discusses our ongoing work.

II. Problem statement

The problem that we are trying to solve is: given a target unicast IP address, t, determine if there is an anycast service being offered via this address, enumerate the replicas that are offering this service, and geolocate those replicas. The means at our disposal are a relatively low number (in the hundreds) of measurement agents situated at m different locations around the world.

We assume that we can launch latency measurements from the vantage points and that we have accurate knowledge of their positions, expressed as latitude and longitude $(\mathrm{lat}, \mathrm{lon})$. Such infrastructures are realistic to obtain, as evidenced by PlanetLab and RIPE Atlas, among others. The geolocation of each vantage point in such services is typically reported by the person who hosts the measurement agent, and these reports can be verified through unicast geolocation services [13]–[17], to check for initial accuracy and to catch cases when an agent is moved to another location.

We use latency measurements to identify discs (circles centered around the vantage points) where anycast replicas lay. While constraint-based geolocation is not a new technique as far as single-host unicast geolocation is concerned, we face a very different problem, as we need to identify and geolocate an unknown number of replicas as opposed to locating a single host. The identification problem is an *optimization problem*, in which an optimal solution consists in identifying all of the replicas. The geolocation problem is a *classification problem*, in which we try to select, from a set of discrete locations within each disc, the most likely position of the anycast replica in that disc.

Our design goals are:

- *completeness*: to fully enumerate anycast replicas
- *accuracy*: to geolocate replicas with city-level precision
- *reliability*: to avoid false positive by design
- *flexibility*: to avoid relying upon or exploiting service- or protocol-specific information
- *low overhead*: to use the smallest possible number of vantage points

Completeness and *accuracy* are obvious goals, necessary to achieve sound results. Sec. V compares the results of our technique to the most recent state of the art [11], [12].

Reliability by design (i.e., absence of false positive detection) is a desirable property that contributes to flexibility of use. For instance, an accurate and lightweight anycast identification technique could be applied to the detection of IP prefix hijacking: it could monitor IP addresses within an address prefix that are expected to be single-host unicast addresses, raising an alarm if anycast behavior is detected. Avoiding false alarms by design would be a necessary condition for such a service.

Flexibility allows the technique to adjust to recent trends in anycast deployment, such as its increasing use for CDNs, and to continue working in the face of new and unprecedented trends. As anycast deployment has until very recently been limited to DNS, it is understandable that that most available techniques are bound to this specific protocol [11], [19], [20] or exploit new developments in DNS [12]. This is in contrast to our protocol- and service- agnostic technique.

Low overhead makes it easier to design and operate continuously running services. Most of the initial studies on DNS anycast characterization [19], [20] relied on a comparable number of vantage points to ours (e.g., about 200 PlanetLab nodes), however more recent studies have employed from 20k vantage points [21] up to 200k recursive DNS resolvers [12] plus 60k Netalyzr datapoints [11]. Our results show this increased overhead to be unjustified, as we achieve similar completeness and accuracy to the most recent state of the art [11], [12] with less than 1/1000th of the vantage points.

III. Methodology

We illustrate our workflow with the help of Fig. 1. From a high level viewpoint, starting from a *dataset of latency measurements* $\mathcal{D}$, that we gather from controlled vantage points (Sec. III-A), we illustrate the anycast detection condition (Sec. III-B). Our *enumeration* approach consists in identifying areas on the globe (the set $\mathcal{E}$) for which we are confident they contain at least one anycast instance (Sec. III-C). For each such area, we then perform a *geolocation* step by identifying the most probable city hosting the instance in a set $\mathcal{G}$ (Sec. III-D). To enhance our coverage, we adopt an *interative* approach, by feeding the solution to the geolocalization $\mathcal{G}(k)$ at step k back to modify the selection of vantage points in the input dataset $\mathcal{D}(k+1)$ at step $k+1$ (Sec. III-E). We now describe each step in more details.

A. Latency measurement

For any given target t, we conduct latency measurements from PlanetLab [22] and from the RIPE infrastructure [23]. For each vantage point p, the measurement infrastructure

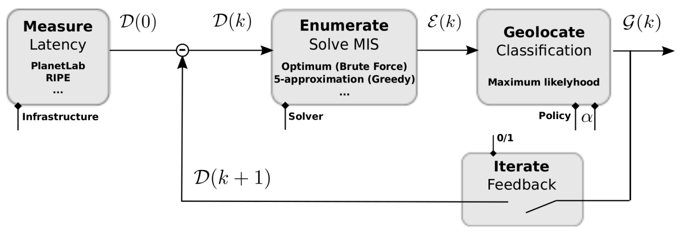

Fig. 1. Methodology workflow

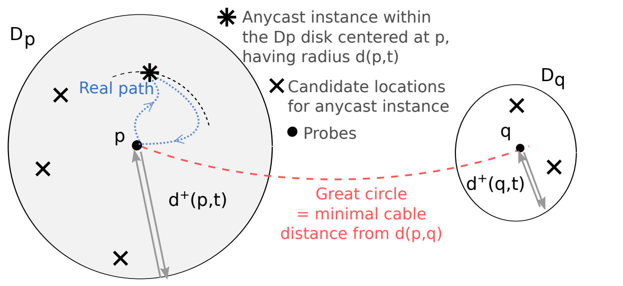
Dp
Anycast instance within the Dp disk centered at p, having radius d(p,t)
Real path
Candidate locations for anycast instance
Probes
Dq
p
q
d+(p,t)
d+(q,t)
Great circle = minimal cable distance from d(p,q)

Fig. 2. Synoptic of anycast instance detection via latency measurements

yields a delay measurement $\delta(p,t)$ representing the round trip propagation delay required for a packet to travel from p to the closest instance of t and back to p.

As an individual latency measurement $\delta_i(p,t)$ can be affected by queuing delay, we estimate $\delta(p,t) = \min_i \delta_i(p,t)/2$ by halving the minimum value of successive round trip time (RTT) measurements (10 in this work). Since network operators generally keep their links to a low load, it is highly probable that at least one of those probes traverses the network without facing congestion in router queues. Although forward and backward paths are not necessarily symmetric, by halving the RTT we make the worst case assumption of maximal distance from the vantage point.

Despite our measurement campaign (Sec. V) includes multiple kind of RTTs measurement (e.g., ICMP, DNS), without loss of generality we focus our attention on ICMP measurements (i.e., the most general) for the remainder of this paper.

B. Anycast Detection

Prior to enumerating anycast instances, we must detect whether there are indeed anycast replicas behind a given unicast IP address. We do so by detecting speed-of-light violations in our dataset by comparing latency measurements δ to the expected propagation time due to speed-of-light considerations. Next, we consider pairs of latency measurements $\delta(p,t)$ and $\delta(q,t)$ for the same target. Specifically, given two vantage points p, q we compute their geodesic distance $d^g(p,q)$ according to Vincenty's formulæ. Since packets cannot travel faster than light, if

$$d^g(p,q) > d^+(p,t) + d^+(q,t) \geq d(p,t) + d(q,t) \qquad (1)$$

then our measurements indicate that p and q are in contact with two separate anycast replicas.

Some remarks are in order. First, (1) compares distances with homogeneous dimensions, that are however gathered with different techniques. Note that considering the geodesic distance $d^g(p,q)$ between vantage points yields a *conservative lower bound* to the expected propagation time between p and q, as a packet will not travel along a geodesic path but will follow a path shaped by physical and economic contraints (i.e., the geography of fiber deployment, optoelectronic conversion, BGP routing, etc.). Conversely, $d^+(p,t) = c_f\delta(p,t)$ *optimistically upper bounds* the distance that a packet may have traveled during $\delta(p,t)$.

As the the inequality is violated only when the conservative lower bound exceeds the optimistic upper bound, it follows (1) is conservative in detecting anycast instances, and by definition avoids raising false positive anycast instances (i.e., flagging as anycast a single-host unicast target).

C. Anycast Enumeration

From a geometric viewpoint, a sufficient condition for (1) to be violated is that the two discs $\mathcal{D}_p$ and $\mathcal{D}_q$ do not overlap. Any time this condition is encountered, we can be certain that there are anycast replicas corresponding to the target unicast IP address t. While this observation is not per se particularly novel [10], we are the first to leverage a full set of distributed measurements in the study of anycast deployment and its geographical properties. Notice that this is extremely important since, while false negatives are possible on a single inequality (i.e., flagging as single-host unicast an anycast target), the chances of a false negative drop with the use of multiple pairs of vantage points.

It is possible that multiple anycast instances may be located within a given disc. Although the aim of anycast is to offer services from distinct locations, the locations may be distinct from an IP routing point of view but not distant geographically from each other. Therefore, our technique can only provide a lower bound of the number of anycast instances that correspond to our observations.

To achieve our joint goals of enumeration and geolocation, we model the problem as a *Maximum Independent Set (MIS)* problem. Our aim is to find a maximum number of vantage points (and corresponding discs) for which we are confident they contact distinct anycast instances (an instance being included in the disc). To do so, we select a maximum subset of discs $\mathcal{E} \subset \mathcal{D}$ such that:

$$\forall \mathcal{D}_p, \mathcal{D}_q \in \mathcal{E}, \qquad \mathcal{D}_p \cap \mathcal{D}_q = \emptyset \qquad (2)$$

The enumeration problem is thus solved by the subset $\mathcal{E}$, whose cardinality $|\mathcal{E}|$ corresponds to the minimum number of instances that avoid latency violations, and which represents thus a plausible explanation to our observations. Notice that $|\mathcal{E}|$ is a lower bound on the number of anycast instances, since due to the conservative definition of (1) we might have removed discs that overlap due to noisy measurements. Additionally, a coarse location of anycast replicas is represented by each disc of $\mathcal{E}$, that constitutes the starting point for the finer grained geolocation of Sec. III-D.

Although the MIS problem is NP-hard, it can be solved in finite time for small number of vantage points with a brute force approach. This allows us to compare the solution of known greedy approximate solutions: while a simple greedy strategy has poor performance in general ($(n-1)-$approximation) the situation improves by simply sorting discs in increasing radius size (5-approximation) as shown in Algorithm 1. We point out that, even though more refined solutions do exist [24] that achieve $(1-\frac{2}{k}) - OPT$ performance, they are however computationally very costly $n^{O(k^4)}$ – and as we will show later in Sec. IV, the greedy solution often performs well in practice.

Algorithm 1 Greedy 5-approximation to MIS for anycast instances enumeration

```
Require: A set of disc $\mathcal{D}$
Ensure: A set of disc $\mathcal{E}$ such as $\forall p, q \in \mathcal{E}, \mathcal{D}_p \cap \mathcal{D}_q = \emptyset$
  Initialization: sort discs in $\mathcal{D}$ by increasing radius size
  Initialization: $\mathcal{E} \leftarrow \emptyset$
  for all disc $\mathcal{D}_d$ of $\mathcal{D}$ do
    for all disc $\mathcal{D}_e$ of $\mathcal{E}$ do
      if $\mathcal{D}_d \cap \mathcal{D}_e = \emptyset$ then
        $\mathcal{E} \leftarrow \mathcal{E} \cup \{\mathcal{D}_e\}$
      end if
    end for
  end for
```

D. Anycast Geolocation

Our aim being to provide geographic locations at city granularity, we need to refine the preliminary location that is output by the enumeration algorithm. We opt for city granularity for two reasons. First, note that a 1 ms difference in latency measurement corresponds to a 100 km disc in geodesic distance terms. It follows that great trust should be put in latency measurements to achieve finer-grained geolocation. Second, notice that ISPs and system administrators often use machine names that map to the city they are serving, which allows us to assess the correctness of our geolocation technique.

As opposed to classical approaches that operate in the geodesic (or Euclidean) space by constructing density maps of likely positions (see references in [15]), or assessing target location to be the center of mass of multiple vantage points [12], we transform the geolocation task into a classification problem as in [14]. Specifically, since our output is a geolocation at city level granularity, we shift the focus from identifying a geographical locus $(lat, lon) \in \mathcal{D}_p \subset \mathbb{R}^2$ to identifying which city $\mathcal{C} \in \mathbf{N}$ contained in the disk $(lat_C, lon_C) \in \mathcal{D}_p$ is most likely hosting the anycast instance.

This focus shift greatly simplifies the problem in two ways: first, it significantly reduces the space cardinality, and, second, it allows us to further leverage additional information with respect to delay or distance measurements, namely the city population. Our reasoning extends previous work [14], which argues that IPs are likely to be located where humans are located: in other words, due to the distribution of population density, large cities represent the likely geolocation of single-host unicast IP addresses. We further argue that, since anycast replicas are specifically engineered to reduce service latency, they ultimately have to be located close to where users live: hence the bias toward large cities is again likely to hold for server side anycast IPs as well.

Our geolocation step outputs IATA airport codes as shorthand for cities. For each of the discs that is output by the enumeration phase, some of the over 7,000 airport codes available worldwide may be contained in the disc. Aside from the trivial case where a single airport is contained in the disc, in the general case multiple airports $\{A_i\} \in \mathcal{D}_p$ are contained in any given disc (represented as a cross in Fig. 2). The output of the geolocation phase can thus be expressed with disc-airport pairs $\mathcal{G} = \{(\mathcal{D}_i, A_i)\}$ according to the notation of Fig. 1.

To guide our selection of the most likely location of a site, we employ two metrics, namely: (i) the population of the main city c_i that the airport A_i serves, for the reasons described above, and (ii) the distance between the airport and the disc border $d(p,t) - d(p, A_i)$, using geographic proximity as a proxy for topological proximity in the routing space.

For a given disc $\mathcal{D}_p$ we compute the likelihood of each airport $\{A_i\} \in \mathcal{D}_p$ for all airports in the disc, as:

$$p_i = \alpha \frac{c_i}{\sum_j c_j} + (1-\alpha) \frac{d(p,t) - d(p,A_i)}{\sum_j d(p,t) - d(p,A_j)} \tag{3}$$

where $p_i \in [0,1]$ follows from the normalization over all airports $\{A_i\} \in \mathcal{D}_p$ of the c_i (population of the main city served by airport A_i) and of the $d(p,t) - d(p,A_i)$ (the distance of the airport i from the disc border) contributions. A parameter $\alpha \in [0,1]$ tunes the relative importance of population vs distance in the decision, in between the distance only ($\alpha = 0$) vs city only ($\alpha = 1$) extremes.

Based on the p_i values, we devise two maximum likelihood policies that return either (i) a single $A_i = \text{argmax}_i p_i$ or (ii) all locations (A_i, p_i) annotated with their respective likelihoods. These policies involve a trade off, as returning all locations increases the average error (since in case $\text{argmax}_i p_i$ is correct, it pays the price of incorrect answers for $1 - p_i$), whereas returning a single location possibly involves a bigger risk.

E. Iteration

Recall that the enumeration step lower bounds the number of instances, due to the possibility of overlapping discs. Now, consider that the geolocation decision in effect transforms a disc $\mathcal{D}_p$, irrespective of its original radius, into a disc $\mathcal{D}'_p$ centered around the selected airport with arbitrarily small radius.

Hence, we argue that, provided the geolocation technique is accurate, it would be beneficial to transform the original set of discs $\mathcal{D}$ by (i) remapping $\mathcal{D}_p$ to $\mathcal{D}'_p$ and (ii) excluding from $\mathcal{D}$ those discs that contain any of the geolocated cities $\mathcal{D}'_p$.

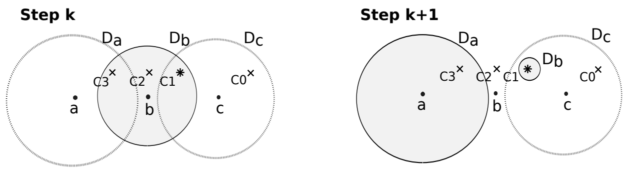

Fig. 3. Synoptic of iterative workflow

Consider for the sake of the example the situation depicted in Fig. 3, where three discs centered around vantage points a, b and c overlap. Let the solution to the enumeration problem at step k select vantage point b (hence the smallest disc $\mathcal{D}_b$), and let furthemore the solution to the geolocation problem select city c_1. By coalescing $\mathcal{D}_b$ around c_1 (with arbitrarily small radius), it follows that at step $k+1$ disk $\mathcal{D}_a$ no longer overlaps with any other disc, meaning that it would be possible to discover another anycast instance (i.e., c_3 in the example) that was previously precluded (whereas disk $\mathcal{D}_c$ still overlap and discovery of c_0 is still precluded).

Denoting with $\mathcal{A}(k)$ the subset of airports geolocated up to step k, and with $\mathcal{G}(k)$ the geolocation at step k (considering a single airport selected per disk for the sake of simplicity):

$$\mathcal{G}(k) = \{(\mathcal{D}_i, A_i) \in \mathcal{E}(k) \times \mathcal{A}(k))\} \tag{4}$$

we have that the dataset $\mathcal{D}(k+1)$ as input to the numeration problem at step $k+1$ would be:

$$\mathcal{D}(k+1) = \mathcal{D}(k) \backslash \{\mathcal{D}_i : \exists (\mathcal{D}_i, A_i) \in \cup_{i=1}^{k} \mathcal{G}(i)\} \tag{5}$$

This workflow can be iterated until no further disc can be added that does not overlap. At each iteration, the set of geolocalized cities grows, so that the set of discs that no longer overlap diminishes, which keep the running time reasonably bounded. Note that iterative operations can be employed irrespectively of the underlying solver (i.e., brute force, greedy, etc.).

IV. Validation

We validate our methodology against publicly available ground truth. For the sake of simplicity, this section considers just 200 PlanetLab vantage points, and defers to Sec. V a more comprehensive study with multiple measurement infrastructures. We first explain our ground truth dataset (Sec. IV-A), then perform a calibration of the enumeration (Sec. IV-B), and the geolocation (Sec. IV-C) tasks.

A. Ground truth

Our technique is not bound to a particular service. However, we are limited in our validation to the availability of reliable ground truth: we thus focus on DNS root servers, 12 of 13 of which use anycast. Indeed, while in general it is challenging to enumerate the sites of an anycast group and associate each site with its geographic location, we are able to build a reliable ground truth for enumeration and geolocation of root servers F, I, K, and L that are operated by ISC, Netnod, RIPE NCC, and ICANN respectively.

TABLE I
Recall of anycast enumeration algorithms with 200 PlanetLab nodes. Gain [%] with respect to the greedy baseline is reported.

	Root server							
Algorithm	**F**		**I**		**K**		**L**	
Greedy	17		13		9		20	
BruteForce	18	+6%	13	-	9	-	20	-
iGreedy	18	+6%	15	+15%	10	+11%	22	+10%
iBruteForce	21	+23%	15	+15%	10	+11%	22	+10%
Dataset CHAOS UB	22		23		11		33	
Published GT	55		46		17		128	

Operators of the root servers maintain an official website [4] with maps annotated with the number and geographic distribution of deployed sites around the world. Additionally we use existing techniques to reliably disambiguate between different instances of the same DNS root server, that exploit queries of the DNS CHAOS class. We stress that CHAOS measurements are only required to assess the enumeration recall and the geolocation accuracy of our proposed methodology (that only relies on distributed delay measurements and is thus not bound to the specific DNS use-case), but are otherwise not used for enumeration and geolocation.

To build a reliable ground truth, we issue distributed IPv4 DNS queries of class CHAOS, type TXT, and name hostname.bind to DNS root servers. Despite the fact that CHAOS replies do not follow a standard format, some operators use IATA airport codes (e.g., AMS, PRG in root severs F and L respectively) and IXPs short names (e.g., AMS-IX, BIX, MIX in root server K) to name their infrastructure servers. In other few cases (e.g., root server I), operators use arbitrary codes, but make publicly available a list that maps site codes to locations. In sporadic cases, multiple CHAOS names are located in the same city: as we are interested in locating geographically distinct anycast replicas, as opposite to enumerating the number of physical or virtual servers operating on a site, we coalesce all replicas located in the same site.

B. Anycast Enumeration

We first benchmark the performance of the anycast enumeration technique. We point out that our aim in this section is not to show the absolute performance (that will be the object of Sec. V), but rather to *relatively compare* the performance of the different solvers (i.e., greedy vs brute force), and to additionally gauge the impact of the iterative workflow.[1]

Under this perspective, Tab. I reports the number of anycast replicas found by solving the maximum independent set problem $|\mathcal{E}|$ for different solvers. Solvers are sorted top to bottom in increasing performance, and the percentage gain with respect to the greedy baseline is also tabulated. For completeness, the table reports the dataset upper bound (UB) and the published ground truth (GT), which represent the number of distinct CHAOS names we were able to observe in our dataset, and the number of distinct servers that are publicly reported on

[1]The performance of the iterative workflow also depends on the geolocation policy: we fix for the time being the *argmax* policy with $\alpha = 0.5$ and justify this choice in Sec. IV-C.

the websites of root servers, respectively. From the table, we gather that the greedy solver achieves in practice performance that is, in most of the cases, as good as that of the brute force solution (I, K, L) or anyway comparable (F). More interestingly, the iterative workflow produces benefits that are sizeable and consistent across datasets and solvers.

Given the strikingly similar recall performance, and considering that the running time of iGreedy (*hundreds of milliseconds*) is orders of magnitude smaller with respect to that of the brute force approach (*hundreds to thousands of seconds*), it seems reasonable to limitedly consider the iterative greedy (iGreedy) solution in what follows. Notice indeed that while we were able to run the brute force solution on the PlanetLab dataset, its cost is prohibitive for larger datasets considered in Sec. V. It also follows that, while refined solutions do exist [24], they are not appealing due to the good recall and short running time of the greedy solver.

Before evaluating the accuracy of our anycast geolocation technique, we proceed with a careful manual validation of the iGreedy enumeration of servers around the world against the published ground truth, and encounter only three spurious cases: one case related to a manifestly wrong location of a PlanetLab node, as well as two cases where airports are located few kilometres away from the discs border (which, as servers are unlikely physically hosted in the airports, is intrisically tied to imprecision due to the naming convention).

C. Anycast Geolocation

We then precisely assess the impact of the geolocation parameters, i.e., the distance vs population weighting factor α and the selection policy argmax vs proportional, by measuring (i) the percentage of correct classification (i.e., geolocation) and (ii) the mean geolocation error in kilometers. For any given disk $\mathcal{D}_p$, let us denote with $A^\star$ the airport code given by the ground truth, and further denote with A_i the different airports that are located in $\mathcal{D}_p$. In case no airport falls in $\mathcal{D}_p$, then we remove the disk, which allows to include further disks at the next iteration. Considering the argmax policy, in case $A^\star = A_i$ (with i such that $\text{argmax}_i\, p_i$), the classification is accounted as correct and the error for this instance is $Err_p = 0\,\text{Km}$. In case $A^\star \neq A_i$, then the classification is erroneous, and off by a distance $Err_p = d(A_i, A^\star)$. In the proportional policy instead, the classification is accounted as correct only for p_i (i.e., proportionally to the percentage of time the correct instance would be selected). The geolocation error for this instance is then computed over all airports inside the disc, and weighted according to the respective likelihood of each airport $Err_p = \sum_j d(A_j, A^\star)p_j$.

Fig. 4 shows the percentage of correct geolocation (left y-axis) and the mean geolocation error (right y-axis) in kilometres for the different policies (argmax vs proportional) and weighting factor (α) under study. Again, in this section we are more interested in the *relative comparison* and the calibration of the technique, rather than absolute performance: under this light, it appears that argmax is preferable to proportional policy. While this does not hold in general, in the iGreedy solution disks having small size are more represented: it follows that distance-based criterion is already fairly accurate, and can be additionally improved by properly taking into account knowledge about city population. Hence, in the remainder of this work we consider an argmax policy that equally weighs distance and population ($\alpha = 0.5$), which leads to jointly high geolocation correctness and low error.

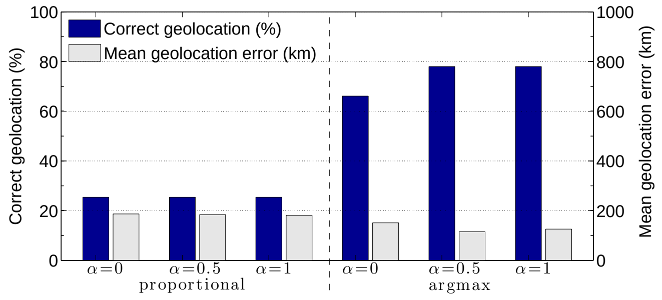

Fig. 4. Geolocation performance with 200 PlanetLab nodes: Percentage of correct geolocation and mean geolocation error for different policies (argmax vs proportional) and weighting factor (α).

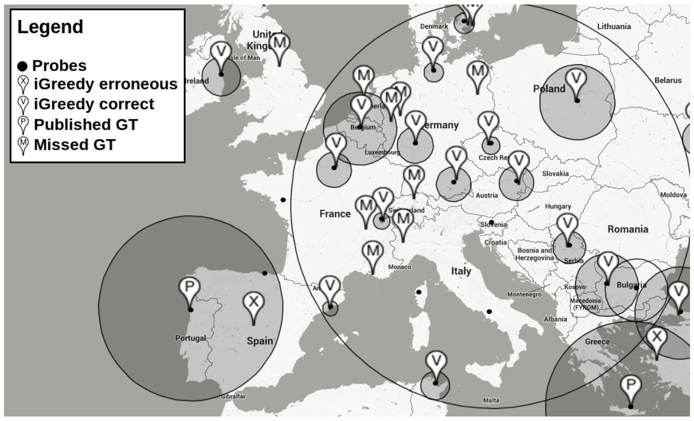

Fig. 6. Enumeration and geolocation of sites for root server L with iGreedy

V. Measurement Campaign

We now report results of our measurement campaign, that include multiple datasets gathered at different times, and using different measurement infrastructure (Sec. V-A). Specifically, our main aim is to critically compare our results to the state of the art (Sec. V-B) and to further assess the robustness of our methodology to vantage point location (Sec. V-C). All our results are browsable at [18].

A. Datasets

We collect datasets from multiple measurement infrastructures (PlanetLab, RIPE) and protocols (ICMP ping, DNS CHAOS application layer measurements). In this section, we consider ICMP measurements over both PlanetLab and RIPE, and build our ground truth as early explained in Sec. IV-A.

In the case of PlanetLab [22], we perform measurements from about 200 nodes located in 26 countries. Node geoloca-

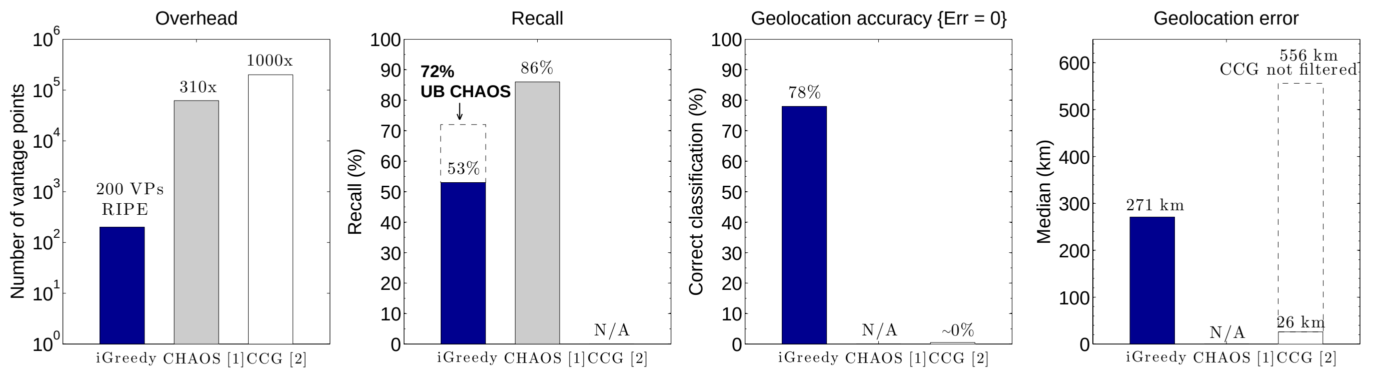

Fig. 5. Performance of iGreedy: Comparison of probing overhead, recall, geolocation accuracy, and geolocation error for misclassified instances with the state of the art [11], [12]. Notice that while [11] are directly *quantitative* comparable, comparison with [12] can be *qualitative* at best.

tion is provided by PlanetLab, and the PlanetLab Europe nodes position is validated with unicast geolocation techniques.

In the case of RIPE [22], we perform measurements from over 6000 nodes located in 350 ASes and 122 countries. Geolocation of RIPE nodes is provided by the users hosting a node. If this information is missing, RIPE finds the location using MaxMind [1] over the IP address of the node.

Since the full RIPE infrastructure comprises 30 times more nodes than PlanetLab, we also consider a subset having approximately the same size, where we select nodes in a stratified sampling as a function of the distance (to ensure geographic vantage points diversity). Unless otherwise stated, results reported in the following are gathered with this dataset, that we denote RIPE200.

B. Comparison against state of the art

1) Eye candy: For the sake of illustration, Fig. 6 reports an example of results for root server L (iGreedy, argmax, $\alpha = 0.5$). The map reports vantage points (black dots) and the results of iGreedy as shaded disks that contain either correct ✓ or erroneous $\times$ geolocation markers (and in the last case the location of the ground truth P as well). The map additionally reports instances that are missed M, either because they are not observed in our measurement, or because they are observed in disks that overlap (represented as circles with no shading).

Several interesting observations are worth making. First, note that despite the very large discs around some vantage points, the MIS formulation factors them out so that no false positives are raised. Second, notice an example of vantage points that our iterative workflow allows to include (i.e., discs of the Bruxelles and Paris vantage points intersect). Third, population bias yields to misclassification for the point located in Porto, Portugal: this vantage point exhibits a relatively large latency (6 ms) to hit a target also located in Porto, so that the disk is large enough to include Madrid (population of 3.3M) which is an order of magnitude more populated than Porto (population of 250K). Refinement of the classification technique is part of our future work (e.g., using logarithmic instead of linear weighting of city population, including distance in the AS space [25], etc.).

2) Comparison at a glance: Fig. 5 summarizes the results discussed in this section. As references for comparison, we take [11], [12]. Notice that the enumeration results of [11] are *directly quantitatively* comparable, as [11] employs F and other root servers as a case study. Geolocation results of the Client-Centric Geolocation (CCG) technique recently proposed in [12] are instead *only qualitatively* comparable, as they target the Google infrastructure. Yet, qualitative comparison allows to gain some interesting methodological insights, that are worth illustrating.

At a glance, Fig. 5 shows that iGreedy (a) reduces the measurement overhead by several orders of magnitude with respect to [11], [12], (b) has comparable yet lower enumeration recall than [11], (c) is able to correctly guess instance location 3/4 of the time unlike [11], [12], (d) has a comparable geolocation error to [12] for the misclassified anycast instances. Not shown in the picture, our methodology is protocol agnostic, unlike [11], [12] that both rely on DNS.

3) Enumeration: Enumeration results are directly comparable, as [11] employs root servers as a case study. We therefore dig further the comparison in Fig. 7. Interestingly, while [11] achieves 100% recall for root servers F and I and 94% and 73% recall for root servers K and L, it does so by issuing DNS CHAOS queries from 62K vantage points (the Netalyzr dataset). In contrast, using only 200 RIPE nodes and the same type of queries, we achieve close recall values for root servers I, K and L (93%, 88%, and 63% respectively). Intuitively, anycast detection relies on the availability of geographically dispersed vantage points. Intrinsically, this means that the datasets used in [11] are highly redundant. Specifically, while the Netalyzr [26] dataset contains over 62k data points, these include possible multiple trials from the same users; similarly, if Netalyzr is popular in a given region, the availability of several points is not useful to increase the overall recall. The same goes for the 200k recursive DNS resolvers exploited by [11], [12] – a clear overkill.

As expected, iGreedy enumerates then only a subset of the CHAOS upper bound (UB) with a recall that remains above 55% (except for root server L). While these results are already satisfactory as they are performed from only a handful of

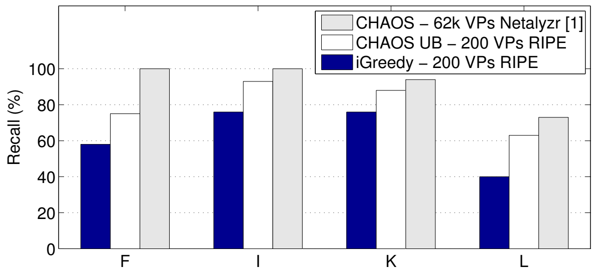

Fig. 7. Enumeration of anycast servers: Comparison of recall with iGreedy over 200 RIPE nodes vs method in [11] from 62k Netalyzr vantage points.

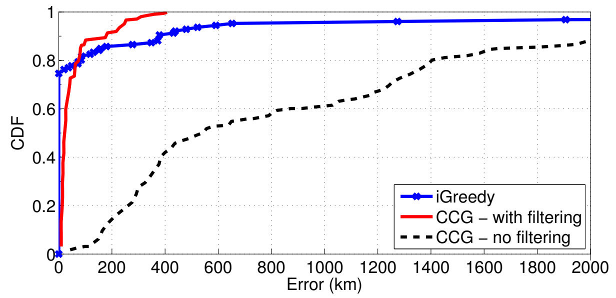

Fig. 8. Geolocation: Comparison of distance error with iGreedy over 200 RIPE nodes vs CCG [12] methodology issuing 4M requests over 200k open recursive DNS resolvers. As previously explained, this comparison has to be interpreted in *qualitative* terms.

vantage points, and additionally allow precise geolocation of the anycast instances, as future work, we aim at increasing iGreedy recall. A promising direction is to merge multiple iGreedy solutions (which is possible due to the very low running time of iGreedy), performed over several selections of fewer vantage points from the same dataset (as chances of overlap reduce, each selection hopefully yields to discovery of some new instances; and since selections are independent, the expected overall recall would increase as well).

4) Geolocation: We finally qualitatively compare geolocation accuracy against CCG [12] in Fig. 8, where it is worth stressing once more that [12] focuses on the geolocation of Google front-end servers, so that the results are not directly comparable in quantitative terms. Yet, as our study is the first to propose anycast geolocation, [12] is the closest technique we can compare with.

First, CCG [12] leverages 200k recursive DNS resolvers to issue over 4M special EDNS-client-subnet queries including IPv4 ranges that would be typical of real Internet clients – hence the Client-Centric Geolocation (CCG) name. Depending on the IPv4 range, the Google DNS server returns a specific replica to be contacted: CCG geolocates such replica in the center of mass of all client IPs directed to this replica, where each client IP is geolocated via MaxMind. As it can be seen from Fig. 8, the CCG has a non negligible error that is intrinsically tied to the accuracy of the MaxMind database [17]. In contrast, in expressing the geolocation task as a classification problem, our technique yields 0 km error for 78% of the enumerated servers and 271 km median error for the misclassified instances.

Additionally, to get rid of noise, CCG proposes to filter out from the cluster vantage points having a distance that exceeds the average by one standard deviation (over each cluster). While this filtering step improves significantly the accuracy of the geolocation, as it can be seen from Fig. 8, iGreedy is still better for the majority of the cases (notice the crossover of the two curves), despite the tail of the error distribution is higher (that could be upper-bounded with filtering, say all circles whose radius exceeds the average by one standard deviation – which we prefer to avoid).

Additionally, CCG filtering also leads to squander of networking and processing resources. In other words, not only are a huge number of vantage points used (200k, which is 1000x more than in our approach), but also quite a significant number of the results from these points (e.g., about 20% or 44k in case delays were normally distributed) are discarded a posteriori. In contrast, our geolocation technique starts from a parsimonious number of vantage points. It then selects only a single reliable vantage point as output of the enumeration phase (i.e., likely among the closest to the instance), after which it performs an informed decision (biased on the city population and the distance from the border).

TABLE II
ROBUSTNESS WITH RESPECT TO VANTAGE POINT SELECTION

	RIPE			PlanetLab
Subset	full	rand	strat	full
Dataset cardinality	6000	500	200	200
iGreedy / CHAOS UB	76%	52%	73%	73%
iGreedy / GT	61%	28%	53%	26%
CHAOS UB / GT	80%	54%	72%	36%
Geolocated	76%	63%	78%	74%
Mean geolocation error (km)	333	569	361	162
iGreedy	149	68	127	65
Dataset CHAOS UB	196	132	174	89
Published GT [F,I,K,L]	246	246	246	246

C. *Robustness with respect to vantage point selection*

Finally, we assess the robustness of the methodology to changes in the vantage points selected. Tab. II summarizes the performance of iGreedy across 4 datasets of different cardinality. Interestingly, we observe that the number of vantage points has little effect on the recall. By applying a naive selection policy on the 6000 RIPE vantage points which consists in selecting 200 vantage points that are at least 100 km distant from each other, we are able to enumerate 127 servers out of the 149 discovered by the full set of vantage points (with a mean geolocation error of 361 km). The recall declines with the PlanetLab dataset which hints that the 200 vantage points selected there do no have good geographical coverage. Clearly, since the number of anycast instances does not exceed 246 for the selected root servers, it is enough to choose hundreds of vantage points that cover the top ASes and the highly populated areas for the enumeration task. A systematic study of vantage point selection is on our research agenda.

VI. Related work

There has been much research on geolocation [13]–[17]. While both measurement-based geolocation techniques [13]–[15] and databases [16], [17] have been thoroughly studied in the unicast field, geolocation of anycast instances is, to the best of our knowledge, a green field so far. While anycast has been the object of much research, most studies either evaluate the performance of IP anycast deployments [20], [21], [27] or propose architectural improvements [20], [28], [29]. Hence, to date only very few studies addressing anycast detection [10], [11], enumeration [11], and especially geolocation [12] of the infrastructure of anycast services exist. Yet, with no exception, despite the fact that anycast is increasingly used for CDNs [9], all the above studies exploit protocol specific information (DNS CHAOS queries in [11] or EDNS extension [12]), unlike our work.

Additionally, a trend towards an increasing number of vantage points was inititated at 20k points [21] and has culminated at 200k recursive DNS resolvers + 60k Netalyzr datapoints [11], [12]. While the earlier work acknowledges that "*measurements from a smaller set of PlanetLab nodes [. . .] were consistent with the larger-scale measurements*" [21], the more recent work states that "*10,000 vantage points are required to reach a recall of 80%*" and that "*90% recall is possible on-demand [...] with 300k recursive DNS servers*" [11]. Our results show this increase to be unjustified.

VII. Conclusion

We propose a novel methodology to reliably and accurately enumerate and geolocate IP-level anycast replicas. An *enumeration* task efficiently solves a maximum independent set optimization problem, and a *geolocation* task successfully exploits side information such as city population to identify the city hosting the instance, feeding back this information in an *iterative* workflow. As our methodology does not rely on protocol specific information, its scope of application is larger than that of existing methodologies. Additionally, as our methodology is lightweight, it enables large scale continuous anycast discovery services. As a side note, due to our design choice to avoid false alarms, the very same service could be useful in detecting IP hijacking. We validate our method and contrast it to state of the art techniques, showing that a handful of vantage points suffices to provide recall [11] and accuracy [12] similar to large-scale techniques.

As we have shown, vantage point location is crucial: we argue that the choice of these points should leverage Internet (e.g., AS number, or IP address) and BGP (e.g., routing distance [25]) knowledge beyond the geographical information exploited in this paper. A related open question concerns the attainable degree of parallelism: i.e., while accuracy is unchanged when 200 vantage points are carefully selected out of 6000, it is unclear how many orthogonal subsets of these 200 points share the same properties. Finally, we argue that an even more efficient approach would be to start from a very small set of vantage points (say, 10) and incrementally add points, using statistical tests on the expected residual number of instances not discovered yet as a stop criterion. These questions are on our research agenda.

Xièxie

This work has been carried out at LINCS (http://www.lincs.fr). The research leading to these results has received funding from the European Union under the FP7 Grant Agreements no. 318627 (Integrated Project "mPlane").

References

[1] https://www.maxmind.com.
[2] C. Partridge, T. Mendez, and W. Milliken, "Host anycasting service," IETF RFC 1546, 1993.
[3] T. Hardie, "Distributing Authoritative Name Servers via Shared Unicast Addresses," IETF RFC 3258, 2002.
[4] http://www.root-servers.org.
[5] https://www.as112.net.
[6] D. Kim, D. Meyer, H. Kilmer, and D. Farinacci, "Anycast Rendevous Point (RP)," RFC 3446, 2003.
[7] C. Huitema, "An Anycast Prefix for 6to4 Relay Routers," IETF RFC 3068, 2001.
[8] B. R. Greene and D. McPherson, "Sink holes: A swiss army knife isp tool," Nanog, 2003.
[9] http://www.edgecast.com.
[10] D. Madory, C. Cook, and K. Miao, "Who are the anycasters," Nanog, 2013.
[11] X. Fan, J. S. Heidemann, and R. Govindan, "Evaluating anycast in the domain name system." in *Proc. IEEE INFOCOM*, 2013.
[12] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann, and R. Govindan, "Mapping the expansion of google's serving infrastructure," in *Proc. ACM IMC*, 2013.
[13] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida, "Constraint-based geolocation of internet hosts." in *Proc. ACM IMC*, 2004.
[14] B. Eriksson, P. Barford, J. Sommers, and R. Nowak, "A learning-based approach for IP geolocation," in *Proc. of PAM*, 2010.
[15] B. Eriksson and M. Crovella, "Understanding geolocation accuracy using network geometry," in *Proc. IEEE INFOCOM*, 2013.
[16] Y. Shavitt and N. Zilberman, "A geolocation databases study," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 10, 2011.
[17] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye, "IP geolocation databases: Unreliable?" *ACM SIGCOMM CCR*, 2011.
[18] http://www.enst.fr/~drossi/anycast.
[19] S. Sarat, V. Pappas, and A. Terzis, "On the use of anycast in DNS," in *Proc. ICCCN*, 2006.
[20] H. Ballani and P. Francis, "Towards a global IP anycast service," in *Proc. ACM SIGCOMM*, 2005.
[21] H. Ballani, P. Francis, and S. Ratnasamy, "A measurement-based deployment proposal for ip anycast." in *Proc. ACM IMC*, 2006.
[22] https://www.planet-lab.org.
[23] https://atlas.ripe.net.
[24] K. Jansen, "Approximation algorithms for geometric intersection graphs," in *Graph-Theoretic Concepts in Computer Science*, 2007.
[25] G. Gürsun, N. Ruchansky, E. Terzi, and M. Crovella, "Routing state distance: A path-based metric for network analysis," in *Proc. ACM IMC*, 2012.
[26] C. Kreibich, N. Weaver, B. Nechaev, and V. Paxson, "Netalyzr: illuminating the edge network," in *Proc. ACM IMC*, 2010.
[27] Z. Liu, B. Huffaker, M. Fomenkov, N. Brownlee, and K. C. Claffy, "Two days in the life of the DNS anycast root servers." in *Proc. of PAM*, 2007.
[28] M. J. Freedman, K. Lakshminarayanan, and D. Mazières, "Oasis: Anycast for any service," in *Proc. USENIX NSDI*, 2006.
[29] D. Katabi and J. Wroclawski, "A framework for scalable global ip-anycast (gia)," in *Proc. ACM SIGCOMM*, 2000.